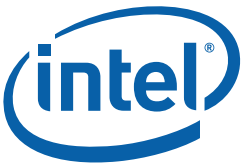


Intel® Celeron® M Processor

Specification Update

February 2008

Revision 039

Document Number: 300303-039

Contents

§

Revision History

Document Number	Revision Number	Description	Date
300303	-001	Initial Release	January 2004
300303	-002	Revisions include: • Added errata WW22- W23 • Added Specification Clarification W1 • Updated Processor Identification table	April 2004
300303	-003	Revisions include: • Added errata WW24- W25	May 2004
300303	-004	Revisions include: • Updated Processor Identification table	June 2004
300303	-005	Revisions include: • Change to Title to reflect processors (adding Celeron M processor ULV on 90-nm processor) • Processor Identification table to include Celeron processor ULV on 90-nm process • Added errata W26 and W27	July 2004
300303	-006	Revisions include: • Updated General information (see Figure 3) for Celeron processor 350 and 360 • Processor Identification table to include Celeron processor 350 and 360	August 2004
300303	-007	Revisions include: • Added errata W28 through W33	November 2004
300303	-008	Revisions include: • Added errata W34 and W35	December 2004
300303	-009	• Updated Processor Identification Table: Added C-0 S-Specs • Updated Summary Tables of Changes • Added Erratum W36 – W38	February 2005
300303	-010	• Updated Processor Identification Table	March 2005

Document Number	Revision Number	Description	Date
300303	-011	• Updated Summary Tables of Changes • Updated Processor Identification Table: — Added Celeron M processor ULV 383 — Updated Celeron M processor 360J & 350J • Added Errata W39 • Added Specification Clarification W1 • Added Specification Clarification W2	May 2005
300303	-012	• Updated Summary Tables of Changes • Removed Erratum W28 – W30 (which were duplicates of W3 – W5) • Added Erratum W40 – W42	June 2005
300303	-013	• Updated Processor Identification Table: o Added Celeron M processor 380	July 2005
300303	-014	• Updated Affected and Related Documents Tables • Updated Processor Identification Table 1	July 2005
300303	-015	• Updated Processor Identification Table 1	October 2005
300303	-016	• Added Erratum W43	November 2005
300303	-017	• Added Errata W44 and W45	December 2005
300303	-018	• Added Errata W46 – W51 • Removed Specification Clarification W1; refer to Section 18.8 of the *IA-32 Intel® Architecture Software Developer's Manual, Volume 3B* for detailed Time Stamp Counter information • Updated Processor Identification Table 1	January 2006
300303	-019	• Added Errata W52 – W66 • Updated Errata W18, W34 • Added General information on Intel Celeron M Processor on 65nm process • Updated Summary of Changes Table to include CPU Signature = 06E8h (Intel Celeron M Processor on 65-nm process) • Updated Processor Identification Table 1 • Updated Affected Documents Table	April 2006
300303	-020	• Updated Errata information as indicated in the Summary Tables of Changes • Updated Processor Identification Table 1 • Added Figure 4	May 2006

Document Number	Revision Number	Description	Date
300303	-021	• Added Errata W83-W87 • Updated Errata W3, W6, W23, W40, W52 • Removed Errata W21, W55, W70, W71, W72, W77 and W81 • Updated Processor Identification (Table 1) • Updated Description for Code "A" and added codes" AA – AE" in Summary Tables of Changes	September 2006
300303	-022	• Added Errata W88-90 • Updated Errata W18, W56, W57, W73 and W80 • Removed erratum W76 • Updated Related Documents Table	September 2006
300303	-023	• Updated Erratum W4	October 2006
300303	-024	• Added Erratum W91	November 2006
300303	-025	• Updated Processor Indentification (Table 1) • Added Errata W92-W134 • Added Intel Celeron M 500 series information and errata • Updated Microcode Update (Table 2)	December 2006
300303	-026	• Updated Errata W26, W32, W56, W59, W62, W65, W67, W68, and W75 • Removed Errata W55, W76, W82 and W91 • Updated Processor Information (Table 1) • Updated Microcode Update (Table 2) • Added Errata W135- W139	January 2007
300303	-027	• Added Errata W140-142	February 2007
300303	-028	• Added Intel Celeron M processor 520 and 530 information • Added Errata W143-W145	March 2007
300303	-029	• Revised Summary Table of Changes • Added Errata W146-W148 • Updated Errata W78	April 2007
300303	-030	• Added Erratum W149 • Added Specification Clarification W3	April 2007 (Out of Cycle)
300303	-031	• Added A-1 Stepping for Celeron M 520 • Updated Errata W53 and W78	May 2007
300303	-032	• Added errata W150 and 151 • Updated Hyperlinks for the SDM Collateral under the "Related Documents" section	June 2007
300303	-033	• Updated Summary Table of Changes	July 2007
300303	-034	• Added erratum W152	August 2007
300303	-035	• Updated Stepping Codes Used in Summary Table • Updated Erratum W41	September 2007

Document Number	Revision Number	Description	Date
300303	-036	• Updated Stepping Codes Used in Summary Table • Added Errata W153 and W154 • Updated Erratum W46	November 2007
300303	-037	• Revised Summary Table of Changes (W75)	December 2007
300303	-038	• Added Erratum W155 • Updated Stepping Codes Used in Summary Table	January 2008
300303	-039	• Updated W11 • Updated W49	February 2008

§

Preface

This document is an update to the specifications contained in the documents listed in the following Affected Documents table. It is a compilation of device and document errata and specification clarifications and changes, and is intended for hardware system manufacturers and for software developers of applications, operating system, and tools.

Information types defined in the Nomenclature section of this document are consolidated into this update document and are no longer published in other documents. This document may also contain information that has not been previously published.

Affected Documents

Document Title	Document Number
Intel® Celeron® M Processor Datasheet	300302-003
Intel® Celeron® M Processor on 65-nm Process Datasheet	312726-004
Intel® Celeron® M Processor on 90-nm Process Datasheet	303110-008

Related Documents

Document Title	Document Number
Intel® 64 and IA-32 Architecture Software Developer's Manual Documentation Changes	252046
Intel® 64 and IA-32 Architecture Software Developer's Manual, Volume 1: Basic Architecture	253665
Intel® 64 and IA-32 Architecture Software Developer's Manual, Volume 2A: Instruction Set Reference, A-M	253666
Intel® 64 and IA-32 Architecture Software Developer's Manual, Volume 2B: Instruction Set Reference, N-Z	253667
Intel® 64 and IA-32 Architecture Software Developer's Manual, Volume 3A: System Programming Guide	253668
Intel® 64 and IA-32 Architecture Software Developer's Manual, Volume 3B: System Programming Guide	253669
IA-32 Intel® Architecture Optimization Reference Manual	248966
Intel Processor Identification and the CPUID Instruction Application Note (AP-485)	241618

Nomenclature

S-Spec Number is a five-digit code used to identify products. Products are differentiated by their unique characteristics, e.g., core speed, L2 cache size, package type, etc. as described in the processor identification information table. Care should be taken to read all notes associated with each S-Spec number.

Errata are design defects or errors. Errata may cause the Intel® Celeron® M processor's behavior to deviate from published specifications. Hardware and software, designed to be used with any given processor, must assume that all errata documented for that processor are present on all devices unless otherwise noted.

Documentation Changes include typos, errors, or omissions from the current published specifications. These changes will be incorporated in the next release of the specifications.

Specification Clarifications describe a specification in greater detail or further highlight a specification's impact to a complex design situation. These clarifications will be incorporated in the next release of the specifications.

Specification Changes are modifications to the current published specifications for the Celeron M processor. These changes will be incorporated in the next release of the specifications.

§

Summary Tables of Changes

The following table indicates the errata, documentation changes, specification clarifications, or specification changes that apply to the Celeron M processor. Intel intends to fix some of the errata in a future stepping of the component and to account for the other outstanding issues through documentation or specification changes as noted. This table uses the following notations:

Codes Used in Summary Table

Stepping

X:	Erratum, Specification Change or Clarification that applies to this stepping.
(No mark) or (Blank Box):	This erratum is fixed in listed stepping or specification change does not apply to listed stepping.

Status

Doc:	Document change or update that will be implemented.
Plan Fix:	This erratum may be fixed in a future of the product.
Fixed:	This erratum has been previously fixed.
No Fix:	There are no plans to fix this erratum.

Row

Shaded:	This item is either new or modified from the previous version of the document.

Note: Each Specification Update item is prefixed with a capital letter to distinguish the product. The key below details the letters that are used in Intel's microprocessor Specification Updates:

A = Dual-Core Intel® Xeon® processor 7000$^{\Delta}$ sequence

C = Intel® Celeron® processor

D = Dual-Core Intel® Xeon® processor 2.80 GHz

E = Intel® Pentium® III processor

F = Intel® Pentium® processor Extreme Edition and Intel® Pentium® D processor

I = Dual-Core Intel® Xeon® processor 5000$^{\Delta}$ series

J = 64-bit Intel® Xeon® processor MP with 1-MB L2 cache

K = Mobile Intel® Pentium® III processor

L = Intel® Celeron® D processor

M = Mobile Intel® Celeron® processor

N = Intel ® Pentium® 4 processor

O = Intel ® Xeon® processor MP

P = Intel ® Xeon® processor

Q = Mobile Intel® Pentium® 4 processor supporting Hyper-Threading Technology on 90-nm process technology

R = Intel® Pentium® 4 processor on 90-nm process

S = 64-bit Intel® Xeon® processor with 800-MHz system bus (1-MB and 2-MB L2 cache versions)

T = Mobile Intel® Pentium® 4 processor–M

U = 64-bit Intel® Xeon® processor MP with up to 8-MB L3 cache

V = Mobile Intel® Celeron® processor on .13 Micron Process in Micro-FCPGA Package

W= Intel® Celeron®-M processor

X = Intel® Pentium® M processor on 90-nm process with 2-MB L2 cache and Intel® Processors A100 and A110 with 512-kB L2 cache

Y = Intel® Pentium® M processor

Z = Mobile Intel® Pentium® 4 processor with 533-MHz system bus

AA= Intel® Pentium® D Processor 900$^{\Delta}$ Sequence and Intel® Pentium® processor Extreme Edition 955$^{\Delta}$, 965$^{\Delta}$

AB= Intel® Pentium® 4 processor 6x1 Sequence

AC= Intel® Celeron® processor in 478-pin package

AD = Intel® Celeron® D processor on 65-nm process

AE = Intel® Core™ Duo processor and Intel® Core™ Solo processor on 65-nm process

AF = Dual-Core™ Intel® Xeon® processor LV

AG = Dual-Core Intel® Xeon® processor 5100$^{\Delta}$ series

AH= Intel® Core™2 Duo/Solo processor for Intel® Centrino® Duo processor technology

AI = Intel® Core™2 Extreme processor X6800$^{\Delta}$ and Intel® Core™2 Duo desktop processor E6000$^{\Delta}$ and E4000$^{\Delta}$ sequence

AJ = Quad-Core Intel® Xeon® processor 5300$^{\Delta}$ series

AK = Intel® Core™2 Extreme quad-core processor QX6000[Δ] sequence and Intel® Core™2 Quad processor Q6000[Δ] sequence

AL = Dual-Core Intel® Xeon® processor 7100 [Δ] series

AM = Intel® Celeron® processor 400 sequence

AN = Intel® Pentium® dual-core processor

AO = Quad-Core Intel® Xeon® processor 3200[Δ] series

AP = Dual-Core Intel® Xeon® processor 3000[Δ] series

AQ = Intel® Pentium® dual-core desktop processor E2000 sequence

AR = Intel® Celeron® Processor 500[Δ] series

AS = Intel® Xeon® processor 7200, 7300 series

AV = Intel® Core™2 Extreme processor QX9650 and Intel® Core™2 Quad processor Q9000 series

AW = Intel® Core™ 2 Duo processor E8000 series

AX = Quad-Core Intel® Xeon® Processor 5400 Series

AY = Dual-Core Intel® Xeon® Processor 5200 Series

AAC = Intel® Celeron® dual-core processor E1000 series

Note: The specification updates for the Pentium® processor, Pentium® Pro processor, and other Intel products do not use this convention.

Note: [Δ] Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. See http://www.intel.com/products/processor_number for details.

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W1			X					No Fix	Performance Monitoring Event That Counts the Number of Instructions Decoded (D0h) Is Not Accurate
W2			X					No Fix	RDTSC Instruction May Report the Wrong Time Stamp Counter Value
W3	X	X	X	X	X	X	X	No Fix	Code Segment Limit Violation May Occur on 4-GB Limit Check

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W4	X	X	X	X	X			No Fix	FST Instruction with Numeric and Null Segment Exceptions May Take Numeric Exception with Incorrect FPU Operand Pointer
W5	X	X	X					No Fix	Code Segment (CS) Is Wrong on SMM Handler When SMBASE Is Not Aligned
W6	X	X	X	X	X			No Fix	A Locked Data Access That Spans across Two Pages May Cause the System to Hang
W7			X					No Fix	Processor Can Enter a Livelock Condition under Certain Conditions When FP Exception Is Pending
W8			X					No Fix	Write Cycle of Write Combining Memory Type Does Not Self Snoop
W9			X					No Fix	Performance Monitoring Event That Counts Floating Point Computational Exceptions (11h) Is Not Accurate.
W10			X					No Fix	Inconsistent Reporting of Data Breakpoints on FP (MMX™ Technology) Loads
W11	X	X	X	X	X	X	X	No Fix	An Enabled Debug Breakpoint or Single Step Trap May Be Taken after MOV SS/POP SS Instruction if it is Followed by an Instruction That Signals a Floating Point Exception
W12			X					No Fix	SysEnter and SysExit Instructions May Write Incorrect Requestor Privilege Level (RPL) in the FP Code Segment Selector (FCS)

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W13	X	X	X	X	X			No Fix	Memory Aliasing with Inconsistent A and D Bits May Cause Processor Deadlock
W14	X	X	X					No Fix	RDMSR or WRMSR to Invalid MSR Address May Not Cause GP Fault
W15			X					No Fix	FP Tag Word Corruption
W16	X	X	X					No Fix	Unable to Disable Reads/Writes to Performance Monitoring Related MSRs
W17	X	X	X					No Fix	Move to Control Register Instruction May Generate a Breakpoint Report
W18	X	X	X	X	X	X	X	No Fix	REP MOVS/STOS Executing with Fast Strings Enabled and Crossing Page Boundaries with Inconsistent Memory Types May Use an Incorrect Data Size or Lead to Memory-Ordering Violations
W19			X					No Fix	The FXSAVE, STOS, or MOVS Instruction May Cause a Store Ordering Violation When Data Crosses a Page with a UC Memory Type
W20			X					No Fix	Machine Check Exception May Occur Due to Improper Line Eviction in the IFU
W21									Errata - Removed
W22			X					No Fix	Performance Event Counter Returns Incorrect Value on L2_LINES_IN Event
W23	X	X	X	X	X	X	X	No Fix	VM Bit Will Be Cleared on Second Fault Handled by Task Switch from Virtual-8086 (VM86)

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W24	X	X	X					No Fix	Code Fetch Matching Disabled Debug Register May Cause Debug Exception
W25	X	X	X					No Fix	Upper Four PAT Entries Not Usable with Mode B or Mode C Paging
W26	X	X	X	X	X			No Fix	SSE/SSE2 Streaming Store Resulting in a Self-Modifying Code (SMC) Event May Cause Unexpected Behavior
W27	X	X						No Fix	Error in Instruction Fetch Unit (IFU) Can Result in an Erroneous Machine Check-Exception (#MC)
W28									Removed, see Erratum W3
W29									Removed, see Erratum W4
W30									Removed, see Erratum W5
W31	X	X	X	X	X	X		No Fix	Page with PAT (Page Attribute Table) Set to USWC (Uncacheable Speculative Write Combine) While Associated MTRR (Memory Type Range Register) Is UC (Uncacheable) May Consolidate to UC
W32	X	X	X	X	X			No Fix	Under Certain Conditions LTR (Load Task Register) Instruction May Result in System Hang
W33	X	X	X					No Fix	Loading from Memory Type USWC (Uncacheable Speculative Write Combine) May Get Its Data Internally Forwarded from a Previous Pending Store

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W34	X	X	X	X	X			No Fix	FPU Operand Pointer May Not Be Cleared Following FINIT/FNINIT
W35	X	X	X					No Fix	FSTP (Floating Point Store) Instruction under Certain Conditions May Result in Erroneously Setting a Valid Bit on an FP (Floating Point) Stack Register
W36		X						No Fix	An Execute Disable Bit Violation May Occur on a Data Page-Fault
W37		X						No Fix	CPUID Leaf 0x80000006 May Provide the Incorrect Value for an 8-Way Associative Cache
W38	X		X					Plan Fix	Snoops during the Execution of a HLT (Halt) Instruction May Lead to Unexpected System Behavior
W39	X	X	X	X	X			No Fix	Invalid Entries in Page-Directory-Pointer-Table Register (PDPTR) May Cause General Protection (#GP) Exception If the Reserved Bits Are Set to One
W40	X	X	X	X	X	X	X	No Fix	INIT Does Not Clear Global Entries in the TLB
W41	X	X	X	X	X	X	X	No Fix	Use of Memory Aliasing with Inconsistent Memory Type May Cause a System Hang or a Machine Check Exception
W42	X	X	X	X	X			No Fix	Machine Check Exception May Occur When Interleaving Code between Different Memory Types

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W43	X	X	X					No Fix	Split I/O Writes Adjacent to Retry of APIC End of Interrupt (EOI) Request May Cause Livelock Condition
W44	X	X	X	X	X	X	X	No Fix	General Protection (#GP) Fault May Not Be Signaled on Data Segment Limit Violation above 4-G Limit
W45	X	X	X	X	X	X	X	No Fix	DR3 Address Match on MOVD/MOVQ/MOVNTQ Memory Store Instruction May Incorrectly Increment Performance Monitoring Count for Saturating SIMD Instructions Retired (Event CFH)
W46	X	X		X	X	X	X	No Fix	Pending x87 FPU Exceptions (#MF) Following STI May Be Serviced before Higher Priority Interrupts
W47	X	X	X					No Fix	Processor INIT# Will Cause a System Hang If Triggered During an NMI Interrupt Routine Performed during Shutdown
W48	X	X	X	X	X			No Fix	Certain Performance Monitoring Counters Related to Bus, L2 Cache and Power Management Are Inaccurate
W49	X	X	X	X	X	X	X	No Fix	Code Segment Limit/Canonical Faults on RSM May be Serviced before Higher Priority Interrupts/Exceptions and May Push the Wrong Address Onto the Stack
W50	X	X	X	X	X	X	X	No Fix	A Write to an APIC Register Sometimes May Appear to Have Not Occurred

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W51	X	X	X	X	X	X	X	No Fix	The Processor May Report a #TS Instead of a #GP Fault
W52	X	X		X	X	X	X	No Fix	BTS Message May Be Lost When the STPCLK# Signal Is Active
W53	X	X		X	X	X	X	No Fix	LER MSRs May be Incorrectly Updated
W54	X	X	X	X	X	X	X	No Fix	Writing the Local Vector Table (LVT) When an Interrupt Is Pending May Cause an Unexpected Interrupt
W55								No Fix	Removed – See erratum W3
W56	X	X	X	X	X	X	X	Plan Fix	FP Inexact-Result Exception Flag May Not Be Set
W57	X	X	X	X	X	X	X	No Fix	MOV To/From Debug Registers Causes Debug Exception
W58				X	X			Plan Fix	Processor Digital Thermal Sensor (DTS) Readout Stops Updating upon Returning from C3 State
W59						x	X	No Fix	LOCK# Asserted during a Special Cycle Shutdown Transaction May Unexpectedly Deassert
W60				X	X	X	X	No Fix	Last Branch Records (LBR) Updates May Be Incorrect after a Task Switch
W61				X	X	X	X	No Fix	Address Reported by Machine-Check Architecture (MCA) on Single-bit L2 ECC Errors May Be Incorrect
W62			X	X	X			No Fix	Disabling of Single-step On Branch Operation May Be Delayed following a POPFD Instruction

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W63				X	X			No Fix	Performance Monitoring Counters That Count External Bus Events May Report Incorrect Values after Processor Power State Transitions
W64				X	X	X	X	No Fix	VERW/VERR/LSL/LAR Instructions May Unexpectedly Update the Last Exception Record (LER) MSR
W65			X	X	X			No Fix	Performance Monitoring Events for Retired Floating Point Operations (C1h) May Not Be Accurate
W66	X	X		X	X	X	X	No Fix	Global Pages in the Data Translation Look-Aside Buffer (DTLB) May Not Be Flushed by RSM Instruction before Restoring the Architectural State from SMRAM
W67			X	X	X			No Fix	Data Breakpoint/Single Step on MOV SS/POP SS May Be Lost after Entry into SMM
W68			X	X	X			No Fix	Hardware Prefetch Performance Monitoring Events May Be Counted Inaccurately
W69				X	X	X	X	No Fix	Programming the Digital Thermal Sensor (DTS) Threshold May Cause Unexpected Thermal Interrupts
W70									Removed - See Erratum W51
W71									Removed - See Erratum W50
W72									Removed - See Erratum W53

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W73	X	X	X	X	X			No Fix	SYSENTER/SYSEXIT Instructions Can Implicitly Load "Null Segment Selector" to SS and CS Registers
W74	X	X	X	X	X	X	X	No Fix	Using 2-M/4-M Pages When A20M# Is Asserted May Result in Incorrect Address Translations
W75			X	X	X		X	No Fix	CPUID Extended Feature Does Not Report Intel® Thermal Monitor 2 Support Correctly
W76									Removed Erratum Due to Redundancy
W77									Removed Erratum Due to Redundancy
W78	X	X		X	X	X	X	No Fix	Premature Execution of a Load Operation Prior to Exception Handler Invocation
W79				X	X	X	X	No Fix	Performance Monitoring Events for Retired Instructions (C0H) May Not Be Accurate
W80	X	X		X	X	X	X	No Fix	#GP Fault Is NOT Generated on Writing IA32_MISC_ENABLE [34] When Execute Disable Bit is Not Supported
W81									Removed Erratum
W82									Removed Erratum- See W26
W83				X	X	X	X	No Fix	Writing Shared Unaligned Data that Crosses a Cache Line without Proper Semaphores or Barriers May Expose a Memory Ordering Issue

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W84				X	X	X	X	No Fix	MSRs Actual Frequency Clock Count (IA32_APERF) or Maximum Frequency Clock Count (IA32_MPERF) May Contain Incorrect Data after a Machine Check Exception (MCE)
W85	X	X	X	X	X	X	X	No Fix	Incorrect Address Computed For Last Byte of FXSAVE/FXRSTOR Image Leads to Partial Memory Update
W86	X	X	X	X	X	X	X	No Fix	Values for LBR/BTS/BTM will be Incorrect after an Exit from SMM
W87				X	X	X	X	No Fix	Using Memory Type Aliasing with Memory Types WB/WT May Lead to Unpredictable Behavior
W88				X	X	X	X	No Fix	Performance Monitoring Event FP_ASSIST May Not be Accurate
W89	X	X	X	X	X	X	X	No Fix	The BS Flag in DR6 May Be Set for Non-Single-Step #DB Exception
W90	X	X		X	X	X	X	No Fix	BTM/BTS Branch-From Instruction Address May Be Incorrect for Software Interrupts
W91									Removed Erratum
W92	x	x	x	x	x	x	X	No Fix	Unaligned Accesses to Paging Structures May Cause the Processor to Hang
W93	x	x	x	x	x	x	X	No Fix	INVLPG Operation for Large (2M/4M) Pages May be Incomplete under Certain Conditions
W94	x	x	x	x	x	x	X	No Fix	Page Access Bit May be Set Prior to Signaling a Code Segment Limit Fault

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W95			x	x	x	x		No Fix	Microcode Updates Performed During VMX Non-root Operation Could Result in Unexpected Behavior
W96				X	X	X		No Fix	Single Step Interrupts with Floating Point Exception Pending May Be Mishandled
W97	X	X	X	X	X	X	X	No Fix	Fault on ENTER Instruction May Result in Unexpected Values on Stack Frame
W98						X	X	No Fix	SYSRET May Incorrectly Clear RF (Resume Flag) in the RFLAGS Register
W99						X	X	No Fix	General Protection Fault (#GP) for Instructions Greater than 15 Bytes May be Preempted
W100						X	X	No Fix	Count Value for Performance-Monitoring Counter PMH_PAGE_WALK May be Incorrect
W101						X	X	No Fix	Performance Monitoring Event For Number Of Reference Cycles When The Processor Is Not Halted (3CH) Does Not Count According To The Specification
W102						X	X	No Fix	Sequential Code Fetch to Non-canonical Address May have Nondeterministic Results
W103						X	X	No Fix	Some Bus Performance Monitoring Events May Not Count Local Events under Certain Conditions
W104						X	X	No Fix	EIP May be Incorrect after Shutdown in IA-32e Mode

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W105						X	X	No Fix	(E)CX May Get Incorrectly Updated When Performing Fast String REP MOVS or Fast String REP STOS With Large Data Structures
W106						X	X	No Fix	Performance Monitoring Events for Retired Loads (CBH) and Instructions Retired (C0H) May Not Be Accurate
W107						X	X	No Fix	Upper 32 bits of 'From' Address Reported through BTMs or BTSs May be Incorrect
W108						X	X	No Fix	Unsynchronized Cross-Modifying Code Operations Can Cause unexpected Instruction Execution Results
W109						X	X	No Fix	Split Locked Stores May not Trigger the Monitoring Hardware
W110						X	X	Plan Fix	REP CMPS/SCAS Operations May Terminate Early in 64-bit Mode when RCX >= 0X100000000
W111						X	X	No Fix	FXSAVE/FXRSTOR Instructions which Store to the End of the Segment and Cause a Wrap to a Misaligned Base Address (Alignment <= 0x10h) May Cause FPU Instruction or Operand Pointer Corruption
W112						X	X	No Fix	PREFETCHh Instruction Execution under Some Conditions May Lead to Processor Livelock
W113						X	X	No Fix	PREFETCHh Instructions May Not be Executed when Alignment Check (AC) is Enabled

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W114						X	X	No Fix	Upper 32 Bits of the FPU Data (Operand) Pointer in the FXSAVE Memory Image May Be Unexpectedly All 1's after FXSAVE
W115						X	X	No Fix	Performance Monitor IDLE_DURING_DIV (18h) Count May Not be Accurate
W116						X	X	No Fix	SYSCALL Immediately after Changing EFLAGS.TF May Not Behave According to the New EFLAGS.TF
W117						X	X	No Fix	IA32_FMASK is Reset during an INIT
W118						X	X	No Fix	IO_SMI Indication in SMRAM State Save Area May Be Set Incorrectly
W119						X	X	No Fix	LBR, BTS, BTM May Report a Wrong Address when an Exception/Interrupt Occurs in 64-bit Mode
W120						X	X	No Fix	A Thermal Interrupt is Not Generated when the Current Temperature is Invalid
W121						X	X	No Fix	CMPSB, LODSB, or SCASB in 64-bit Mode with Count Greater or Equal to 248 May Terminate Early
W122						X	X	No Fix	Returning to Real Mode from SMM with EFLAGS.VM Set May Result in Unpredictable System Behavior
W123						X	X	No Fix	IRET under Certain Conditions May Cause an Unexpected Alignment Check Exception

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W124						X	X	No Fix	CPL-Qualified BTS May Report Incorrect Branch-From Instruction Address
W125						X	X	No Fix	PEBS Does Not Always Differentiate Between CPL-Qualified Events
W126						X	X	No Fix	PMI May Be Delayed to Next PEBS Event
W127						X	X	No Fix	PEBS Buffer Overflow Status Will Not be Indicated Unless IA32_DEBUGCTL[12] is Set
W128						X	X	No Fix	An Asynchronous MCE During a Far Transfer May Corrupt ESP
W129						X	X	No Fix	B0-B3 Bits in DR6 May Not be Properly Cleared After Code Breakpoint
W130						X	X	No Fix	REP Store Instructions in a Specific Situation may cause the Processor to Hang
W131						X	X	No Fix	Performance Monitor SSE Retired Instructions May Return Incorrect Values
W132						X	X	No Fix	Performance Monitoring Events for L1 and L2 Miss may Not be Accurate
W133						X		No Fix	A MOV Instruction from CR8 Register with 16-Bit Operand Size Will Leave Bits 63:16 of the Destination Register Unmodified
W134						X	X	No Fix	Debug Register May Contain Incorrect Information on a MOVSS or POPSS Instruction Followed by SYSRET

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W135				x	x	X	X	Plan Fix	Update of Read/Write (R/W) or User/Supervisor (U/S) or Present (P) Bits without TLB Shootdown May Cause Unexpected Processor Behavior
W136						X	X	Plan Fix	Update of Attribute Bits on Page Directories without Immediate TLB Shootdown May Cause Unexpected Processor Behavior
W137						X	X	Plan Fix	Invalid Instructions May Lead to Unexpected Behavior
W138	x	x	x	x	x	X	X	No Fix	EFLAGS, CR0, CR4 and the EXF4 Signal May be Incorrect after Shutdown
W139						X	X	Plan Fix	Performance Monitoring Counter MACRO_INSTS.DECODED May Not Count Some Decoded Instructions
W140						X	X	Plan Fix	The Stack May be Incorrect as a Result of VIP/VIF Check on SYSEXIT and SYSRET
W141						X	X	No Fix	Performance Monitoring Event SIMD_UOP_TYPE_EXEC.MUL is Counted Incorrectly for PMULUDQ Instruction
W142						X	X	No Fix	Storage of PEBS Record Delayed Following Execution of MOV SS or STI
W143	X	X	X	X	X	X	X	No Fix	Store Ordering May be Incorrect between WC and WP Memory Types
W144						X	X	No Fix	Updating Code Page Directory Attributes without TLB Invalidation May Result in Improper Handling of Code #PF

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W145						X		Plan Fix	Performance Monitoring Event CPU_CLK_UNHALTED.REF May Not Count Clock Cycles According to the Processors Operating Frequency
W146						X	X	Plan Fix	Performance Monitoring Event BR_INST_RETIRED May Count CPUID Instructions as Branches
W147						X	X	No Fix	Performance Monitoring Event MISALIGN_MEM_REF May Over Count
W148						X	X	No Fix	A REP STOS/MOVS to a MONITOR/MWAIT Address Range May Prevent Triggering of the Monitoring Hardware
W149						X	X	Plan Fix	False Level One Data Cache Parity Machine-Check Exceptions May be Signaled
W150						X	X	No Fix	A Memory Access May Get a Wrong Memory Type Following a #GP due to WRMSR to an MTRR Mask
W151						X	X	No Fix	PMI While LBR Freeze Enabled May Result in Old/Out-of-date LBR Information
W152	X	X	X	X	X	X	X	No Fix	Performance Monitoring Event FP_MMX_TRANS_TO_MMX May Not Count Some Transitions
W153						X	X	No Fix	A WB Store Following a REP STOS/MOVS of FXSAVE May Lead to Memory-Ordering Violations

NO.	Steppings							Plans	ERRATA
	B-1 CPU Signature = 06D6h	C-0 CPU Signature = 06D8h	B-1 CPU Signature = 0695h	C-0 CPU Signature = 06E8h	D-0 CPU Signature = 06ECh	B-2 CPU Signature = 06F6h	A-1 CPU Signature = 10661h		
W154							X	No Fix	RSM Instruction Execution under Certain Conditions May Cause Processor Hang or Unexpected Instruction Execution Results
W155	X	X	X	X	X	X	X	No Fix	Benign Exception after a Double Fault May Not Cause a Triple Fault Shutdown

Number	SPECIFICATION CHANGES
	There are no Specification Changes in this Specification Update revision

Number	SPECIFICATION CLARIFICATIONS
W1	Specification Clarification with Respect to Time-Stamp Counter - Removed
W2	Thermal Diode Offset Specification Clarification - Removed
W3	Clarification of TRANSLATION LOOKASIDE BUFFERS (TLBS) Invalidation

Number	DOCUMENTATION CHANGES
	There are no Documentation Changes in this Specification Update revision

§

Identification Information

The Celeron M processor can be identified by the following values:

Family[1]	Model[2]	Model[2] (Celeron® M on 90-nm process)	Brand ID [3]
0110	1001	1101	00010010

NOTES:

1. The Family corresponds to bits [11:8] of the EDX register after Reset, bits [11:8] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register.
2. The Model corresponds to bits [7:4] of the EDX register after Reset, bits [7:4] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register.
3. The Brand ID corresponds to bits [7:0] of the EBX register after the CPUID instruction is executed with a 1 in the EAX register.

Table 1. Intel Celeron M Processor Identification

S-Spec	Processor number	Product Stepping	L2 Cache Size (bytes)	CPU Signature	Core Frequency	Bus Frequency	Voltage	Package Micro-FCBGA-Pb= Micro-FCBGA Lead Free
SL9WN	520	A-1	1 M	10011b	1.60	533 MHz	1.30 V-0.95 V	Micro-FCPGA
SL9VA	530	B-2	1 M	06F6h	1.73	533 MHz	1.30 V- 0.95 V	Micro-FCPGA
SL9WT	520	B-2	1 M	06F6h	1.60	533 MHz	1.30 V-0.95 V	Micro-FCPGA
SL9KX	450	D-0	1 M	06ECh	2.00	533 MHz	1.30 V-1.00 V	Micro-FCPGA
SL9KW	440	D-0	1 M	06ECh	1.86	533 MHz	1.30 V-1.00 V	Micro-FCPGA
SL8W2	410	C-0	1 M	06E8h	1.46	533 MHz	1.30-1.00 V	Micro-FCPGA
SL92F	430	C-0	1 M	06E8h	1.73	533 MHz	1.30-1.00 V	Micro-FCPGA
SL8VZ	420	C-0	1 M	06E8h	1.60	533 MHz	1.30-1.00 V	Micro-FCPGA
SL9KV	430	D-0	1 M	06ECh	1.73	533 MHz	1.30-1.00 V	Micro-FCPGA
SL8XW	423	C-0	1 M	06E8h	1.06	533 MHz	1.10-0.85 V	Micro-FCBGA
SL8MH	390	C-0	1 M	06D8h	1.7 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA
SL8MP	390	C-0	1 M	06D8h	1.7 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA
SL8MV	390	C-0	1 M	06D8h	1.7 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA-Pb
SL8LP	383	C-0	1 M	06D8h	1.00 GHz	400 MHz	0.876 V – 0.956 V[1]	Micro-FCBGA
SL8LV	383	C-0	1 M	06D8h	1.00 GHz	400 MHz	0.876 V – 0.956 V[1]	Micro-FCBGA-Pb
SL8BP	383	C-0	1 M	06D8h	1.00 GHz	400 MHz	0.940 V	Micro-FCBGA
SL8BN	383	C-0	1 M	06D8h	1.00 GHz	400 MHz	0.940 V	Micro-FCBGA-Pb
SL8LQ	373	C-0	512 K	06D8h	1.00 GHz	400 MHz	0.876 V – 0.956 V[1]	Micro-FCBGA

S-Spec	Processor number	Product Stepping	L2 Cache Size (bytes)	CPU Signature	Core Frequency	Bus Frequency	Voltage	Package Micro-FCBGA-Pb= Micro-FCBGA Lead Free
SL8LW	373	C-0	512 K	06D8h	1.00 GHz	400 MHz	0.876 V – 0.956 V[1]	Micro-FCBGA-Pb
SL8A4	373	C-0	512 K	06D8h	1.00 GHz	400 MHz	0.940 V	Micro-FCBGA
SL89S	373	C-0	512 K	06D8h	1.00 GHz	400 MHz	0.940 V	Micro-FCBGA-Pb
SL8MN	380	C-0	1 M	06D8h	1.60 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCPGA
SL8MG	380	C-0	1 M	06D8h	1.60 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA
SL8MU	380	C-0	1 M	06D8h	1.60 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA-Pb
SL8MM	370	C-0	1 M	06D8h	1.50 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCPGA
SL8MF	370	C-0	1 M	06D8h	1.50 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA
SL8MT	370	C-0	1 M	06D8h	1.50 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA-Pb
SL86J	370	C-0	1 M	06D8h	1.50 GHz	400 MHz	1.260 V	Micro-FCPGA
SL86P	370	C-0	1 M	06D8h	1.50 GHz	400 MHz	1.260 V	Micro-FCBGA
SL86D	370	C-0	1 M	06D8h	1.50 GHz	400 MHz	1.260 V	Micro-FCBGA-Pb
SL8ML	360J	C-0	1 M	06D8h	1.40 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCPGA
SL8ME	360J	C-0	1 M	06D8h	1.40 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA
SL86K	360J	C-0	1 M	06D8h	1.40 GHz	400 MHz	1.260 V	Micro-FCPGA
SL86Q	360J	C-0	1 M	06D8h	1.40 GHz	400 MHz	1.260 V	Micro-FCBGA
SL8MK	350J	C-0	1 M	06D8h	1.30 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCPGA
SL8MD	350J	C-0	1 M	06D8h	1.30 GHz	400 MHz	1.004 V – 1.292 V[1]	Micro-FCBGA
SL86L	350J	C-0	1 M	06D8h	1.30 GHz	400 MHz	1.260 V	Micro-FCPGA
SL86R	350J	C-0	1 M	06D8h	1.30 GHz	400 MHz	1.260 V	Micro-FCBGA
SL7LS	360	B-1	1 M	06D6h	1.4 GHz	400 MHz	1.260 V	Micro-FCPGA
SL7LR	360	B-1	1 M	06D6h	1.4 GHz	400 MHz	1.260 V	Micro-FCBGA
SL7RA	350	B-1	1 M	06D6h	1.3 GHz	400 MHz	1.260 V	Micro-FCPGA
SL7R9	350	B-1	1 M	06D6h	1.3 GHz	400 MHz	1.260 V	Micro-FCBGA
SL6N7	320	B-1	512 K	0695h	1.30 GHz	400 MHz	1.356 V	Micro-FCPGA
SL6NM	320	B-1	512 K	0695h	1.30 GHz	400 MHz	1.356 V	Micro-FCBGA
SL6N6	330	B-1	512 K	0695h	1.40 GHz	400 MHz	1.356 V	Micro-FCPGA
SL6NL	330	B-1	512 K	0695h	1.40 GHz	400 MHz	1.356 V	Micro-FCBGA
SL7MT	340	B-1	512 K	0695h	1.50 GHz	400 MHz	1.356 V	Micro-FCBGA
SL7ME	340	B-1	512 K	0695h	1.50 GHz	400 MHz	1.356 V	Micro-FCPGA
SL79S	n/a	B-1	512 K	0695h	1.20 GHz	400 MHz	1.356 V	Micro-FCPGA
SL79T	n/a	B-1	512 K	0695h	1.20 GHz	400 MHz	1.356 V	Micro-FCBGA

S-Spec	Processor number	Product Stepping	L2 Cache Size (bytes)	CPU Signature	Core Frequency	Bus Frequency	Voltage	Package Micro-FCBGA-Pb= Micro-FCBGA Lead Free
SL7GE	n/a	B-1	512 K	0695h	600 MHz	400 MHz	1.004 V	Micro-FCBGA
SL7DB	n/a	B-1	512 K	0695h	800 MHz	400 MHz	1.004 V	Micro-FCBGA
SL7DH	n/a	B-1	512 K	0695h	900 MHz	400 MHz	1.004 V	Micro-FCBGA
SL7F7	353	B-1	512 K	06D6h	900 MHz	400 MHz	0.940 V	Micro-FCBGA
SL7QX	353	B-1	512 K	06D6h	900 MHz	400 MHz	0.940 V	Micro-FCBGA-Pb

NOTES:

1. These are optimized Voltage ID (VID) values. Individual processor VID values may be calibrated during manufacturing such that two devices at the same speed may have different VID settings.

Component Marking Information

Figure 1. Intel Celeron M Processor (Micro-FCPGA/FCBGA) S-Spec Markings

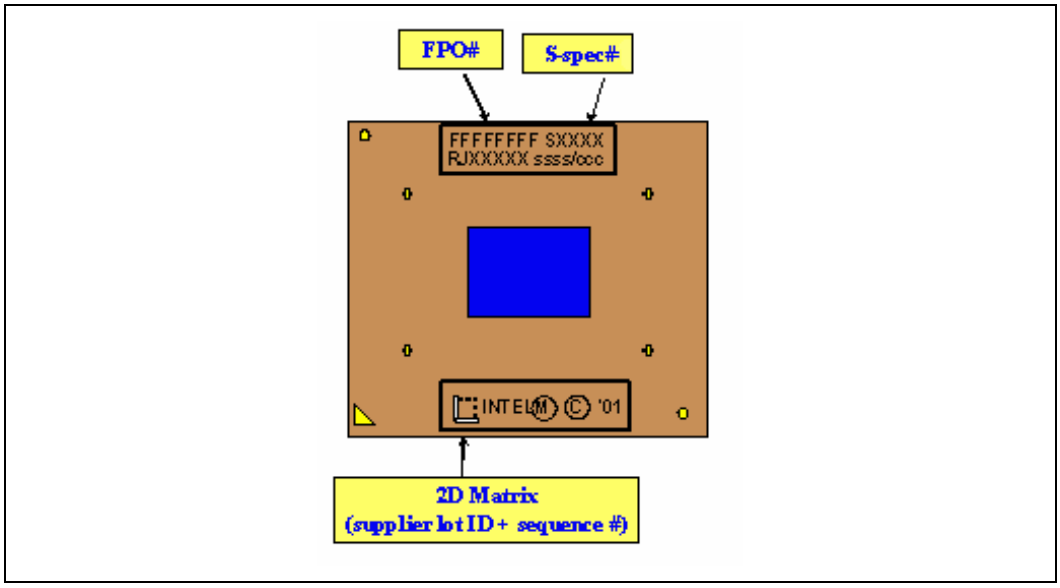


Figure 2. Intel Celeron Processor ULV on 90-nm Process (Micro-FCBGA) S-Spec Markings

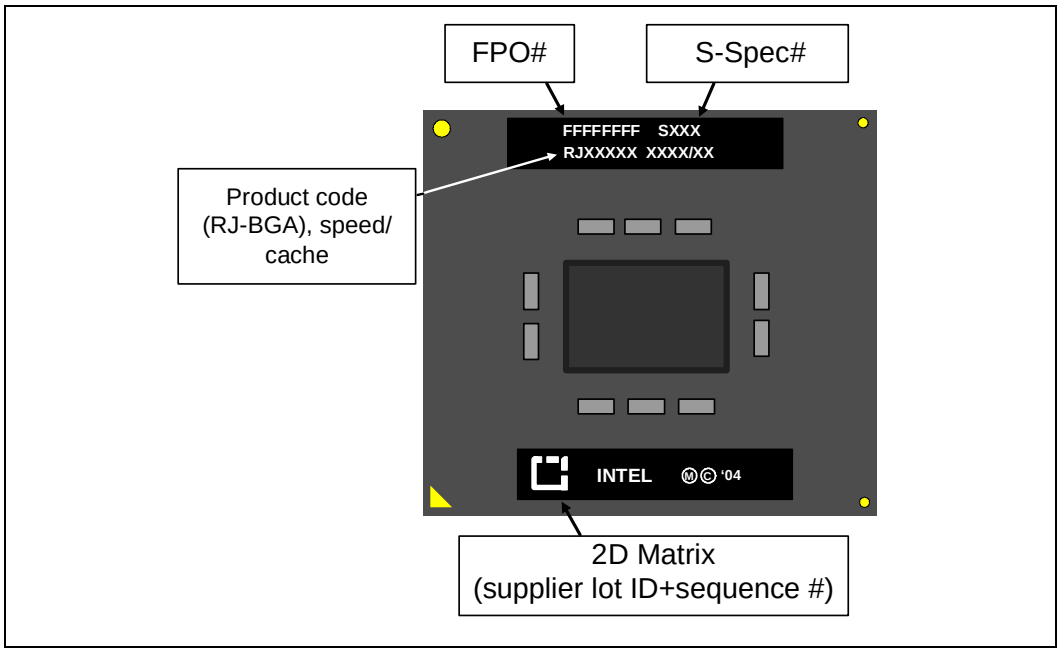


Figure 3. Intel Celeron M Processor on 90-nm Process (Micro-FCPGA/FCBGA) S-Spec Markings

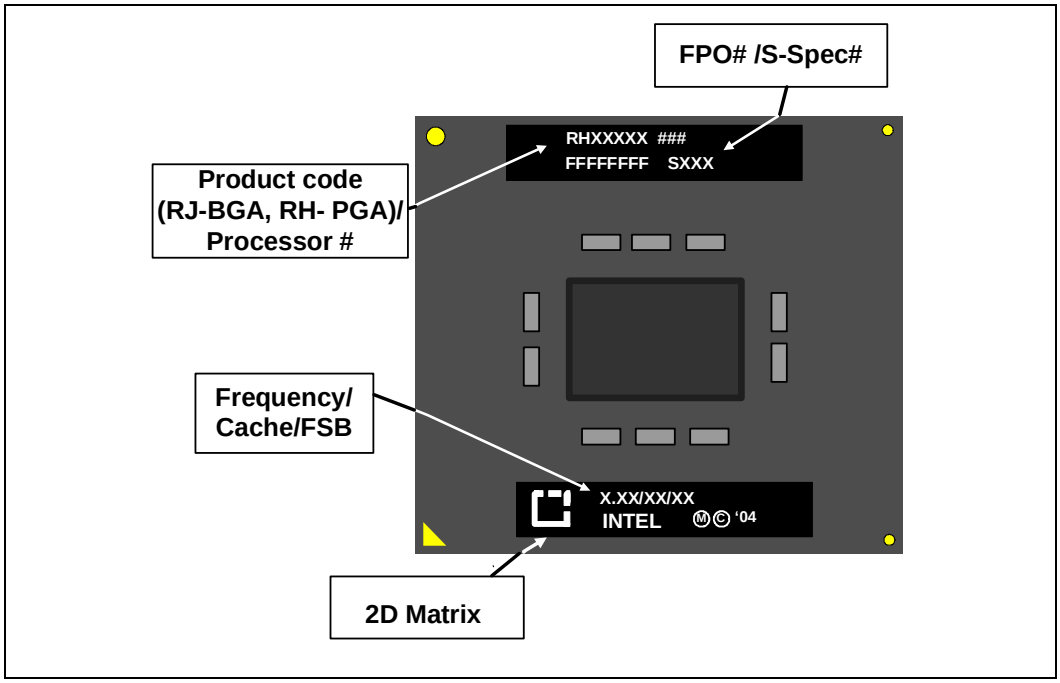
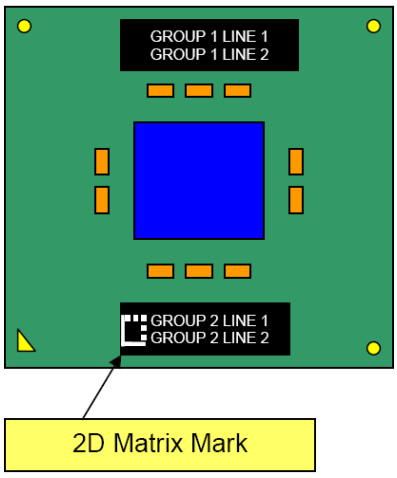


Figure 4. Intel Celeron M Processor on 65-nm Process (Micro-FCPGA/FCBGA) S-Spec Markings



PRODUCTION MARK EXAMPLE:
GROUP 1 LINE 1: Unique Unit Identifier [Processor #]
GROUP 1 LINE 2: FPO S-SPEC#
GROUP 2 LINE 1: FREQUENCY/ L2 CACHE SIZE/ FSB
GROUP 2 LINE 1: INTEL m © '04

§

Errata

W1. Performance Monitoring Event That Counts the Number of Instructions Decoded (D0h) Is Not Accurate

Problem: The performance-monitoring event that counts the number of instructions decoded may have inaccurate results.

Implication: There is no functional impact of this erratum. However, the results/counts from this performance monitoring event should not be considered as being accurate

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W2. RDTSC Instruction May Report the Wrong Time-Stamp Counter Value

Problem: The time-stamp counter is a 64-bit counter that is read in two, 32-bit chunks. The counter incorrectly advances and therefore the two chunks may go out of synchronization causing the Read Time-stamp Counter (RDTSC) instruction to report the wrong time-stamp counter value.

Implication: This erratum may cause software to see the wrong representation of processor time and may result in unpredictable software operation.

Workaround: It is possible for BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W3. Code Segment Limit Violation May Occur on 4-GB Limit Check

Problem: Code Segment limit violation may occur on 4-GB limit check when the code stream wraps around in a way that one instruction ends at the last byte of the segment and the next instruction begins at 0x0.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially-available software, or system.

Workaround: Avoid code that wraps around segment limit.

Status: For the steppings affected, see the Summary Tables of Changes.

W4. FST Instruction with Numeric and Null Segment Exceptions May Take Numeric Exception with Incorrect FPU Operand Pointer

Problem: If execution of an FST (Store Floating Point Value) instruction would generate both numeric and null segment exceptions, the numeric exception may be taken first and with the Null x87 FPU Instruction Operand (Data) Pointer.

Implication: Due to this erratum, on an FST instruction the processor reports a numeric exception instead of reporting an exception because of a Null segment. If the numeric exception handler tries to access the FST data it will get a #GP fault. Intel has not observed this erratum with any commercially available software, or system.

Workaround: The numeric exception handler should check the segment and if it is Null avoid further access to the data that caused the fault.

Status: For affected steppings see the Summary Table of Changes.

W5. Code Segment (CS) Is Wrong on SMM Handler When SMBASE Is Not Aligned

Problem: With SMBASE being relocated to a non-aligned address, during SMM entry the CS can be improperly updated which can lead to an incorrect SMM handler.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially-available software, or system.

Workaround: Align SMBASE to 32 kB.

Status: For the steppings affected, see the Summary Tables of Changes.

W6. A Locked Data Access That Spans across Two Pages May Cause the System to Hang

Problem: An instruction with lock data access that spans across two pages may, given some rare internal conditions, hang the system.

Implication: When this erratum occurs, the system may hang. Intel has not observed this erratum with any commercially-available software or system.

Workaround: A locked data access should always be aligned.

Status: For the steppings affected, see the Summary Tables of Changes.

W7. Processor Can Enter a Livelock Condition under Certain Conditions When FP Exception Is Pending

Problem: Processor clock modulation may be controlled via a processor register (IA32_THERM_CONTROL) or via the STPCLK# signal. While the processor clock is constantly being actively modulated at 12.5% and 25% duty cycles and there is a pending unmasked FP exception (ES pending), if you attempt a FP load (or MMX™ technology Mov

instruction) and the load has an longer than typical latency the processor can enter a livelock.

Implication: When this erratum occurs, the processor will enter a livelock condition. Intel has not observed this erratum with any commercially-available software or system.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W8. Write Cycle of Write Combining Memory Type Does Not Self Snoop

Problem: Write cycles of WC memory type do not self-snoop. This may result in data inconsistency- if the addresses of the WC data are aliased to WB memory type memory, which has been cached. In such a case, the internal caches will not be updated with the WC data sent on the system bus.

Implication: This condition may result in a data inconsistency. Intel has not observed this erratum with any commercially-available software, system, nor components.

Workaround: Software should detect via the self-snoop bit in the CPUID features flags if the processor supports a self-snooping capability. Software should perform explicit memory management/flushing for aliased memory ranges on processor that do not self-snoop.

Status: For the steppings affected, see the Summary Tables of Changes.

W9. Performance Monitoring Event That Counts Floating Point Computational Exceptions (11h) Is Not Accurate

Problem: Performance monitoring event that counts Floating Point Compare exceptions may have inaccurate results.

Implication: There is no functional impact of this erratum. However this Performance Monitoring Event should not be used when accurate performance monitoring is required.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W10. Inconsistent Reporting of Data Breakpoints on FP (MMX Technology) Loads

Problem: The reporting of data breakpoints on either FP or MMX technology loads is dependent upon the code faulting behavior prior to the execution of the load. If there is a fault pending prior to the execution of the load and FP exceptions are enabled there is a chance that data breakpoint on successive FP/MMX technology Loads may be reported twice.

Implication: Software debuggers should be aware of this possibility. There should be no implications to software operated outside of a debug environment.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W11. An Enabled Debug Breakpoint or Single Step Trap May Be Taken after MOV SS/POP SS Instruction if it is Followed by an Instruction That Signals a Floating Point Exception

Problem: A MOV SS/POP SS instruction should inhibit all interrupts including debug breakpoints until after execution of the following instruction. This is intended to allow the sequential execution of MOV SS/POP SS and MOV [r/e]SP, [r/e]BP instructions without having an invalid stack during interrupt handling. However, an enabled debug breakpoint or single step trap may be taken after MOV SS/POP SS if this instruction is followed by an instruction that signals a floating point exception rather than a MOV [r/e]SP, [r/e]BP instruction. This results in a debug exception being signaled on an unexpected instruction boundary since the MOV SS/POP SS and the following instruction should be executed atomically.

Implication: This can result in incorrect signaling of a debug exception and possibly a mismatched Stack Segment and Stack Pointer. If MOV SS/POP SS is not followed by a MOV [r/e]SP, [r/e]BP, there may be a mismatched Stack Segment and Stack Pointer on any exception. Intel has not observed this erratum with any commercially available software, or system.

Workaround: As recommended in the IA32 Intel® Architecture Software Developer's Manual, the use of MOV SS/POP SS in conjunction with MOV [r/e]SP, [r/e]BP will avoid the failure since the MOV [r/e]SP, [r/e]BP will not generate a floating point exception. Developers of debug tools should be aware of the potential incorrect debug event signaling created by this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W12. SysEnter and SysExit Instructions May Write Incorrect Requestor Privilege Level (RPL) in the FP Code Segment Selector (FCS)

Problem: SysEnter and SysExit instructions may write incorrect RPL in the FP Code Segment selector (FCS). As a result of this, the RPL field in FCS may be corrupted.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially-available software, or system.

Workaround: It is possible for BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W13. Memory Aliasing with Inconsistent A and D Bits May Cause Processor Deadlock

Problem: In the event that software implements memory aliasing by having two-page directory entries (PDEs) point to a common page table entry (PTE) and the Accessed and Dirty bits for the two PDEs are allowed to become inconsistent, the processor may become deadlocked.

Implication: This erratum has not been observed with commercially-available software.

Workaround: Software that needs to implement memory aliasing in this way should manage the consistency of the Accessed and Dirty bits.

Status: For the steppings affected, see the Summary Tables of Changes.

W14. RDMSR or WRMSR to Invalid MSR Address May Not Cause GP Fault

Problem: The RDMSR and WRMSR instructions allow reading or writing of MSR's (Model Specific Registers) based on the index number placed in ECX. The processor should reject access to any reserved or unimplemented MSRs by generating #GP(0). However, there are some invalid MSR addressers for which the processor will not generate #GP(0). This erratum has not been observed with commercially-available software.

Implication: For RDMSR, undefined values will be read into EDX:EAX. For WRMSR, undefined processor behavior may result.

Workaround: Do not use invalid MSR addresses with RDMSR or WRMSR.

Status: For the steppings affected, see the Summary Tables of Changes.

W15. FP Tag Word Corruption

Problem: In some rare cases, fault information generated as the result of instruction execution may be incorrect. The result is an incorrect FP stack entry.

Implication: This erratum may result in corruption of the FP tag word in a way that a non-valid entry in the FP stack may become valid. The software is not expected to read a non-valid entry. If the software attempts to use the stack entry (which is expected to be empty) the result may be an erroneous "Stack overflow."

Workaround: Do not disable SSE/SSE2 in control register CR4 and avoid code segment limit violation.

Status: For the steppings affected, see the Summary Tables of Changes.

W16. Unable to Disable Reads/Writes to Performance Monitoring Related MSRs

Problem: The Performance Monitoring Available bit in the miscellaneous processor features MSR (IA32_MISC_ENABLES.7) was defined so that

when it is cleared to a 0, RDMSR/WRMSR/RDPMC instructions would return all zeros for reads of and prevent any writes to performance monitoring related MSRs. Currently it is possible to read from or write to performance monitoring related MSRs when the Performance Monitoring Available bit is cleared to a 0.

Implication: It is not possible to disallow reads and writes to the Performance Monitoring MSRs. Intel has not observed this erratum with any commercially-available software or system.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W17. Move to Control Register Instruction May Generate a Breakpoint Report

Problem: A move (MOV) to control register (CR) instruction where control register is CR0, CR3 or CR4 may generate a breakpoint report.

Implication: MOV to control register instruction is not expected to generate a breakpoint report.

Workaround: Ignore breakpoint data from MOV to CR instruction.

Status: For the steppings affected, see the Summary Tables of Changes.

W18. REP MOVS/STOS Executing with Fast Strings Enabled and Crossing Page Boundaries with Inconsistent Memory Types May Use an Incorrect Data Size or Lead to Memory-Ordering Violations

Problem: Under certain conditions as described in the *Intel® 64 and IA-32 Architecture Software Developer's Manual* section, *Out-of-Order Stores For String Operations in Pentium 4, Intel Xeon, and P6 Family Processors*, the processor performs REP MOVS or REP STOS as fast strings. Due to this erratum fast string REP MOVS/REP STOS instructions that cross page boundaries from WB/WC memory types to UC/WP/WT memory types, may start using an incorrect data size or may observe memory ordering violations.

Implication: Upon crossing the page boundary the following may occur, dependent on the new page memory type:

- UC the data size of each write will now always be 8 bytes, as opposed to the original data size.
- WP the data size of each write will now always be 8 bytes, as opposed to the original data size and there may be a memory ordering violation.
- WT there may be a memory ordering violation.

Workaround: Software should avoid crossing page boundaries from WB or WC memory type to UC, WP or WT memory type within a single REP MOVS or REP STOS instruction that will execute with fast strings enabled.

Status: For the steppings affected, see the Summary Tables of Changes.

W19. The FXSAVE, STOS, or MOVS Instruction May Cause a Store Ordering Violation When Data Crosses a Page with a UC Memory Type

Problem: If the data from an FXSAVE, STOS, or MOVS instruction crosses a page boundary from WB to UC memory type and this instruction is immediately followed by a second instruction that also issues a store to memory, the final data stores from both instructions may occur in the wrong order.

Implication: The impact of this store ordering behavior may vary from normal software execution to potential software failure. Intel has not observed this erratum in commercially-available software.

Workaround: FXSAVE, STOS, or MOVS data must not cross page boundary from WB to UC memory type.

Status: For the steppings affected, see the Summary Tables of Changes.

W20. Machine Check Exception May Occur Due to Improper Line Eviction in the IFU

Problem: The processor is designed to signal an unrecoverable machine check exception (MCE) as a consistency checking mechanism. Under a complex set of circumstances involving multiple speculative branches and memory accesses there exists a one cycle long window in which the processor may signal a MCE in the instruction fetch unit (IFU) because instructions previously decoded have been evicted from the IFU. The one cycle long window is opened when an opportunistic fetch receives a partial hit on a previously executed but not as yet completed store resident in the store buffer. The resulting partial hit erroneously causes the eviction of a line from the IFU at a time when the processor is expecting the line to still be present. If the MCE for this particular IFU event is disabled, execution will continue normally.

Implication: While this erratum may occur on a system with any number of processors, the probability of occurrence increases with the number of processors. If this erratum does occur, a machine check exception will result. Note systems that implement an operating system that does not enable the Machine Check Architecture will be completely unaffected by this erratum (e.g., Microsoft Windows* 95 and Windows*98).

Workaround: It is possible for BIOS code to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W21. POPF and POPFD Instructions That Set the Trap Flag Bit May Cause Unpredictable Processor Behavior

Problem: In some rare cases, POPF and POPFD instructions that set the Trap Flag (TF) bit in the EFLAGS register (causing the processor to enter Single-Step mode) may cause unpredictable processor behavior.

Implication: Single step operation is typically enabled during software debug activities, not during normal system operation.

Workaround: There is no workaround for single step operation in commercially-available software. For debug activities on custom software the POPF and POPFD instructions could be immediately followed by a NOP instruction to facilitate correct execution.

Status: For the steppings affected, see the Summary Tables of Changes.

W22. Performance Event Counter Returns Incorrect Value on L2_LINES_IN Event

Problem: The performance event counter returns an incorrect value on L2_LINES_IN event (EMON event #24H) when the L2 cache is disabled.

Implication: Due to this erratum, L2_LINES_IN performance event counter should not be monitored while the L2 cache is disabled. This erratum has no functional impact.

Workaround: Ignore L2_LINES_IN event when the L2 cache is disabled.

Status: For the steppings affected, see the Summary Tables of Changes.

W23. VM Bit Is Cleared on Second Fault Handled by Task Switch from Virtual-8086 (VM86)

Problem: Following a task switch to any fault handler that was initiated while the processor was in VM86 mode, if there is an additional fault while servicing the original task switch then the VM bit will be incorrectly cleared in EFLAGS, data segments will not be pushed and the processor will not return to the correct mode upon completion of the second fault handler via IRET.

Implication: When the OS recovers from the second fault handler, the processor will no longer be in VM86 mode. Normally, operating systems should prevent interrupt task switches from faulting, thus the scenario should not occur under normal circumstances.

Workaround: None identified.

Status: For the steppings affected, see the see the Summary Tables of Changes.

W24. Code Fetch Matching Disabled Debug Register May Cause Debug Exception

Problem: The bits L0-3 and G0-3 enable breakpoints local to a task and global to all tasks, respectively. If one of these bits is set, a breakpoint is enabled, corresponding to the addresses in the debug registers DR0-DR3. If at least one of these breakpoints is enabled, any of these

registers are disabled (i.e., L*n* and G*n* are 0), and RW*n* for the disabled register is 00 (indicating a breakpoint on instruction execution), normally an instruction fetch will not cause an instruction-breakpoint fault based on a match with the address in the disabled register(s). However, if the address in a disabled register matches the address of a code fetch which also results in a page fault, an instruction-breakpoint fault will occur.

Implication: While debugging software, extraneous instruction-breakpoint faults may be encountered if breakpoint registers are not cleared when they are disabled. Debug software which does not implement a code breakpoint handler will fail, if this occurs. If a handler is present, the fault will be serviced. Mixing data and code may exacerbate this problem by allowing disabled data breakpoint registers to break on an instruction fetch.

Workaround: The debug handler should clear breakpoint registers before they become disabled.

Status: For the steppings affected, see the Summary Tables of Changes.

W25. Upper Four PAT Entries Not Usable with Mode B or Mode C Paging

Problem: The page attribute table (PAT) contains eight entries, which must all be initialized and considered when setting up memory types for the Pentium III processor. However, in Mode B or Mode C paging, the upper four entries do not function correctly for 4-Kbyte pages. Specifically, bit 7 of page table entries that translate addresses to 4-Kbyte pages should be used as the upper bit of a 3-bit index to determine the PAT entry that specifies the memory type for the page. When Mode B (CR4.PSE = 1) and/or Mode C (CR4.PAE) are enabled, the processor forces this bit to zero when determining the memory type regardless of the value in the page table entry. The upper four entries of the PAT function correctly for 2-Mbyte and 4-Mbyte large pages (specified by bit 12 of the page directory entry for those translations).

Implication: Only the lower four PAT entries are useful for 4-kB translations when Mode B or C paging is used. In Mode A paging (4-Kbyte pages only), all eight entries may be used. All eight entries may be used for large pages in Mode B or C paging.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W26. SSE/SSE2 Streaming Store Resulting in a Self-Modifying Code (SMC) Event May Cause Unexpected Behavior

Problem: An SSE or SSE2 streaming store that results in a self-modifying code (SMC) event may cause unexpected behavior. The SMC event occurs on a full address match of code contained in L1 cache.

Implication: Due to this erratum, any of the following events may occur:

1. A data access break point may be incorrectly reported on the instruction pointer (IP) just before the store instruction.
2. A non-cacheable store can appear twice on the external bus (the first time it will write only 8 bytes, the second time it will write the entire 16 bytes).

Intel has not observed this erratum with any commercially-available software.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W27. Error in Instruction Fetch Unit (IFU) Can Result in an Erroneous Machine Check-Exception (#MC)

Problem: A rare combination of events including the generation of a bus lock(s), the execution of a WBINVD instruction, and a page accessed or dirty bit assist may result in an erroneous Machine Check-Exception (#MC).

Implication: Due to this erratum, unexpected machine check-exception (#MC) is generated. Intel has not been able to reproduce this erratum with commercially-available software.

Workaround: It is possible for BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W28. Removed; See Erratum W3.

W29. Removed; See Erratum W4.

W30. Removed; See Erratum W5.

W31. Page with PAT (Page Attribute Table) Set to USWC (Uncacheable Speculative Write Combine) While Associated MTRR (Memory Type Range Register) Is UC (Uncacheable) May Consolidate to UC

Problem: A page whose PAT memory type is USWC while the relevant MTRR memory type is UC, the consolidated memory type may be treated as UC (rather than WC, as specified in the *Intel® 64 and IA-32 Architecture Software Developer's Manual*).

Implication: When this erratum occurs, the memory page may be as UC (rather than WC). This may have a negative performance impact.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W32. Under Certain Conditions LTR (Load Task Register) Instruction May Result in System Hang

Problem: A LTR instruction may result in a system hang if all the following conditions are met:

1. Invalid data selector of the TR (Task Register) resulting with either #GP (General Protection Fault) or #NP (Segment Not Present Fault).
2. GDT (Global Descriptor Table) is not 8-bytes aligned.
3. Data BP (breakpoint) is set on cache line containing the descriptor data.

Implication: This erratum may result in system hang if all conditions have been met. This erratum has not been observed in commercial operating systems or software. For performance reasons, GDT is typically aligned to 8-bytes.

Workaround: Align GDT to 8 bytes.

Status: For the steppings affected, see the Summary Tables of Changes.

W33. Loading from Memory Type USWC (Uncacheable Speculative Write Combine) May Get Its Data Internally Forwarded from a Previous Pending Store

Problem: A load from memory type USWC may get its data internally forwarded from a pending store. As a result, the expected load may never be issued to the external bus.

Implication: When this erratum occurs, a USWC load request may be satisfied without being observed on the external bus. There are no known usage models where this behavior results in any negative side-effects.

Workaround: Do not use memory type USWC for memory that has read side-effects.

Status: For the steppings affected, see the Summary Tables of Changes.

W34. FPU Operand Pointer May Not Be Cleared following FINIT/FNINIT

Problem: Initializing the floating point state with either FINIT or FNINT, may not clear the x87 FPU Operand (Data) Pointer Offset and the x87 FPU Operand (Data) Pointer Selector (both fields form the FPUDataPointer). Saving the floating point environment with FSTENV, FNSTENV, or floating point state with FSAVE, FNSAVE or FXSAVE before an intervening FP instruction may save uninitialized values for the FPUDataPointer.

Implication: When this erratum occurs, the values for FPUDataPointer in the saved floating point image structure may appear to be random values. Executing any non-control FP instruction with memory operand will initialize the FPUDataPointer. Intel has not observed this erratum with any commercially-available software.

Workaround: After initialization, do not expect a floating point state saved memory image to be correct, until at least one non-control FP instruction with a memory operand has been executed.

Status: For the steppings affected, see the Summary Tables of Changes.

W35. FSTP (Floating Point Store) Instruction Under Certain Conditions May Result in Erroneously Setting a Valid Bit on an FP (Floating Point) Stack Register

Problem: When an FSTP instruction with a PDE/PTE (Page Directory Entry/Page Table Entry) A/D bit update is followed by a user mode access fault due to a code fetch to a page that has supervisor only access permission, this may result in erroneously setting a valid bit of an FP stack register. The FP top of stack pointer is unchanged.

Implication: This erratum may cause an unexpected stack overflow.

Workaround: User mode code should not depend on being able to recover from illegal accesses to memory regions protected with supervisor only access when using FP instructions.

Status: For the steppings affected, see the Summary Tables of Changes.

W36. An Execute Disable Bit Violation May Occur on a Data Page-Fault

Problem: Under a combination of internal events, unexpected Execute Disable violations may occur on data accesses that are Execute Disable protected.

Implication: This erratum may cause unexpected Execute Disable violations.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W37. CPUID Leaf 0x80000006 May Provide the Incorrect Value for an 8-Way Associative Cache

Problem: CPUID leaf 0x80000006 may return 0x8 in ECX [15:12] to indicate 8-way associative cache, but the correct encoding for an 8-way associative cache is 0x6.

Implication: Software that depends on the associativity of the cache may not function correctly.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W38. Snoops during the Execution of a HLT (Halt) Instruction May Lead to Unexpected System Behavior

Problem: If during the execution of a HLT instruction an external snoop causes an eviction from the instruction fetch unit (IFU) instruction cache, the processor may, on exit from the HLT state, erroneously read stale data from the victim cache.

Implication: This erratum may lead to unexpected system behavior. Intel has only observed this condition in non-mobile configurations.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W39. Invalid Entries in Page-Directory-Pointer-Table Register (PDPTR) May Cause General Protection (#GP) Exception If the Reserved Bits Are Set to One

Problem: Invalid entries in the Page-Directory-Pointer-Table Register (PDPTR) that have the reserved bits set to one may cause a General Protection (#GP) exception.

Implication: Intel has not observed this erratum with any commercially-available software.

Workaround: Do not set the reserved bits to one when PDPTR entries are invalid.

Status: For the steppings affected, see the Summary Tables of Changes.

W40. INIT Does Not Clear Global Entries in the TLB

Problem: INIT may not flush a TLB entry when:

1. The processor is in protected mode with paging enabled and the page global enable flag is set (PGE bit of CR4 register)
2. G bit for the page table entry is set
3. TLB entry is present in TLB when INIT occurs

Implication: Software may encounter unexpected page fault or incorrect address translation due to a TLB entry erroneously left in TLB after INIT.

Workaround: Write to CR3, CR4 (setting bits PSE, PGE or PAE) or CR0 (setting bits PG or PE) registers before writing to memory early in BIOS code to clear all the global entries from TLB.

Status: For the steppings affected, see the Summary Tables of Changes.

W41. Use of Memory Aliasing with Inconsistent Memory Type May Cause a System Hang or a Machine Check Exception

Problem: Software that implements memory aliasing by having more than one linear addresses mapped to the same physical page with different cache types may cause the system to hang or to report a Machine Check Exception (MCE). This would occur if one of the addresses is non-cacheable used in code segment and the other a cacheable address. If the cacheable address finds its way in instruction cache, and non-cacheable address is fetched in IFU, the processor may invalidate the non-cacheable address from the fetch unit. Any micro-architectural event that causes instruction restart will expect this instruction to still be in fetch unit and lack of it will cause a system hang or an MCE.

Implication: This erratum has not been observed with commercially-available software.

Workaround: Although it is possible to have a single physical page mapped by two, different linear addresses with different memory types, Intel has strongly discouraged this practice as it may lead to undefined results. Software that needs to implement memory aliasing should manage the memory type consistency.

Status: For the steppings affected, see the Summary Tables of Changes.

W42. Machine Check Exception May Occur When Interleaving Code between Different Memory Types

Problem: A small window of opportunity exists where code fetches interleaved between different memory types may cause a machine check exception. A complex set of micro-architectural boundary conditions is required to expose this window.

Implication: Interleaved instruction fetches between different memory types may result in a machine check exception. The system may hang if machine check exceptions are disabled. Intel has not observed the occurrence of this erratum while running commercially-available applications or operating systems.

Workaround: Software can avoid this erratum by placing a serializing instruction between code fetches between different memory types.

Status: For the steppings affected, see the Summary Tables of Changes.

W43. Split I/O Writes Adjacent to Retry of APIC End of Interrupt (EOI) Request May Cause Livelock Condition

Problem: When Split I/O instruction writes occur adjacent to a retry of a Local APIC End of Interrupt (EOI) request by the chipset, a livelock condition may result. The required sequences of events are:

1. The processor issues a Local APIC EOI message.
2. The chipset responds with a retry because its downstream ports are full. It expects the processor to return with the same EOI request.
3. The processor issues a Split I/O write instruction instead.
4. The chipset responds with a retry because it expected the APIC EOI.
5. The processor insists the Split I/O write instruction must be completed and issues write instruction again.

Implication: A processor livelock may occur causing a system hang. This issue has only been observed in synthetic lab testing conditions and has not been seen in any commercially-available applications. The erratum does not occur with Intel mobile chipset-based platforms.

Workaround: Use the PIC instead of the APIC for the interrupt controller.

Status: For the steppings affected, see the Summary Tables of Changes.

W44. General Protection (#GP) Fault May Not Be Signaled on Data Segment Limit Violation above 4-G Limit

Problem: Memory accesses to flat data segments (base = 00000000h) that occur above the 4-G limit (0ffffffffh) may not signal a #GP fault.

Implication: When such memory accesses occur, the system may not issue a #GP fault.

Workaround: Software should ensure that memory accesses do not occur above the 4-G limit (0ffffffffh).

Status: For the steppings affected, see the Summary Tables of Changes.

W45. DR3 Address Match on MOVD/MOVQ/MOVNTQ Memory Store Instruction May Incorrectly Increment Performance Monitoring Count for Saturating SIMD Instructions Retired (Event 0CFh)

Problem: Performance monitoring for Event CFH normally increments on saturating SIMD instruction retired. Regardless of DR7 programming, if the linear address of a retiring memory store MOVD/MOVQ/MOVNTQ instruction executed matches the address in DR3, the CFH counter may be incorrectly incremented.

Implication: The value observed for performance monitoring count for saturating SIMD instructions retired may be too high. The size of error is dependent on the number of occurrences of the conditions described above, while the counter is active.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W46. Pending x87 FPU Exceptions (#MF) following STI May Be Serviced before Higher Priority Interrupts

Problem: Interrupts that are pending prior to the execution of the STI (Set Interrupt Flag) instruction are normally serviced immediately after the instruction following the STI. An exception to this is if the following instruction triggers a #MF. In this situation, the interrupt should be serviced before the #MF. Because of this erratum, if following STI, an instruction that triggers a #MF is executed while STPCLK#, Enhanced Intel SpeedStep Technology transitions or Thermal Monitor events occur, the pending #MF may be serviced before higher priority interrupts.

Implication: Software may observe #MF being serviced before higher priority interrupts.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W47. Processor INIT# Will Cause a System Hang If Triggered during an NMI Interrupt Routine Performed during Shutdown

Problem: During the execution of an NMI interrupt handler, if shutdown occurs followed by the INIT# signal being triggered, the processor will attempt initialization but fail soft reset.

Implication: Due to this erratum the system may hang.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W48. Certain Performance Monitoring Counters Related to Bus, L2 Cache and Power Management Are Inaccurate

Problem: All Performance Monitoring Counters in the ranges 21H-3DH and 60H-7FH may have inaccurate results up to ±7.

Implication: There may be a small error in the affected counts.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W49. Code Segment Limit/Canonical Faults on RSM May be Serviced before Higher Priority Interrupts/Exceptions and May Push the Wrong Address Onto the Stack

Problem: Normally, when the processor encounters a Segment Limit or Canonical Fault due to code execution, a #GP (General Protection Exception) fault is generated after all higher priority Interrupts and exceptions are serviced. Due to this erratum, if RSM (Resume from System Management Mode) returns to execution flow that results in a Code Segment Limit or Canonical Fault, the #GP fault may be serviced before a higher priority Interrupt or Exception (e.g. NMI (Non-Maskable Interrupt), Debug break(#DB), Machine Check (#MC), etc.). If the RSM attempts to return to a non-canonical address, the address pushed onto the stack for this #GP fault may not match the non-canonical address that caused the fault.

Implication: Operating systems may observe a #GP fault being serviced before higher priority Interrupts and Exceptions. Intel has not observed this erratum on any commercially available software.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W50. A Write to an APIC Register Sometimes May Appear to Have Not Occurred

Problem: With respect to the retirement of instructions, stores to the uncacheable memory-based APIC register space are handled in a non-synchronized way. For example, if an instruction that masks the interrupt flag (e.g., CLI) is executed soon after an uncacheable write

to the Task Priority Register (TPR) that lowers the APIC priority, the interrupt masking operation may take effect before the actual priority has been lowered. This may cause interrupts whose priority is lower than the initial TPR, but higher than the final TPR, to not be serviced until the interrupt enabled flag is finally set, i.e., by STI instruction. Interrupts will remain pending and are not lost.

Implication: In this example the processor may allow interrupts to be accepted but may delay their service.

Workaround: This non-synchronization can be avoided by issuing an APIC register read after the APIC register write. This will force the store to the APIC register before any subsequent instructions are executed. No commercial operating system is known to be impacted by this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W51. The Processor May Report a #TS Instead of a #GP Fault

Problem: A jump to a busy TSS (Task-State Segment) may cause a #TS (invalid TSS exception) instead of a #GP fault (general protection exception).

Implication: Operation systems that access a busy TSS may get invalid TSS fault instead of a #GP fault. Intel has not observed this erratum with any commercially-available software.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W52. BTS Message May Be Lost When the STPCLK# Signal Is Active

Problem: STPCLK# is asserted to enable the processor to enter a low-power state. Under some circumstances, when STPCLK# becomes active, a pending BTS (Branch Trace Store) message may be either lost and not written or written with corrupted branch address to the Debug Store area.

Implication: BTS messages may be lost in the presence of STPCLK# assertions.

Workaround: None identified.

Status: For affected steppings see the Summary Tables of Changes.

W53. Last Exception Record (LER) MSRs May Be Incorrectly Updated

Problem: The LASTINTTOIP and LASTINTFROMIP MSRs (1DDH-1DEH) may contain incorrect values after the following events: masked SSE2 floating-point exception, StopClk, NMI and INT.

Implication: The value of the LER MSR may be incorrectly updated to point to a SIMD Floating-Point instruction even though no exception occurred on that instruction or to point to an instruction that was preceded by a

StopClk interrupt or rarely not to be updated on Interrupts (NMI and INT).

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W54. Writing the Local Vector Table (LVT) When an Interrupt Is Pending May Cause an Unexpected Interrupt

Problem: If a local interrupt is pending when the LVT entry is written, an interrupt may be taken on the new interrupt vector even if the mask bit is set.

Implication: An interrupt may immediately be generated with the new vector when an LVT entry is written, even if the new LVT entry has the mask bit set. If there is no Interrupt Service Routine (ISR) set up for that vector the system will GP fault. If the ISR does not do an End of Interrupt (EOI) the bit for the vector will be left set in the in-service register and mask all interrupts at the same or lower priority.

Workaround: Any vector programmed into an LVT entry must have an ISR associated with it, even if that vector was programmed as masked. This ISR routine must do an EOI to clear any unexpected interrupts that may occur. The ISR associated with the spurious vector does not generate an EOI; therefore the spurious vector should not be used when writing the LVT.

Status: For the steppings affected, see the Summary Tables of Changes.

W55. Removed – See Erratum W3

W56. FP Inexact-Result Exception Flag May Not Be Set

Problem: When the result of a floating-point operation is not exactly represented in the destination format (1/3 in binary form, for example), an inexact-result (precision) exception occurs. When this occurs, the PE bit (bit 5 of the FPU status word) is normally set by the processor. Under certain rare conditions, this bit may not be set when this rounding occurs. However, other actions taken by the processor (invoking the software exception handler if the exception is unmasked) are not affected. This erratum can only occur if the floating-point operation which causes the precision exception is immediately followed by one of the following instructions:

- FST m32real
- FST m64real
- FSTP m32real
- FSTP m64real
- FSTP m80real
- FIST m16int

- FIST m32int
- FISTP m16int
- FISTP m32int
- FISTP m64int

Note that even if this combination of instructions is encountered, there is also a dependency on the internal pipelining and execution state of both instructions in the processor.

Implication: Inexact-result exceptions are commonly masked or ignored by applications, as it happens frequently, and produces a rounded result acceptable to most applications. The PE bit of the FPU status word may not always be set upon receiving an inexact-result exception. Thus, if these exceptions are unmasked, a floating-point error exception handler may not recognize that a precision exception occurred. Note that this is a "sticky" bit, i.e., once set by an inexact-result condition, it remains set until cleared by software.

Workaround: This condition can be avoided by inserting two non-floating-point instructions between the two floating-point instructions.

Status: For the steppings affected, see the Summary Tables of Changes.

W57. MOV to/from Debug Registers Causes Debug Exception

Problem: When in V86 mode, if a MOV instruction is executed to/from a debug register, a general-protection exception (#GP) should be generated. However, in the case when the general detect enable flag (GD) bit is set, the observed behavior is that a debug exception (#DB) is generated instead.

Implication: With debug-register protection enabled (i.e., the GD bit set), when attempting to execute a MOV on debug registers in V86 mode, a debug exception will be generated instead of the expected general-protection fault.

Workaround: In general, operating systems do not set the GD bit when they are in V86 mode. The GD bit is generally set and used by debuggers. The debug exception handler should check that the exception did not occur in V86 mode before continuing. If the exception did occur in V86 mode, the exception may be directed to the general-protection exception handler.

Status: For the steppings affected, see the Summary Tables of Changes.

W58. Processor Digital Thermal Sensor (DTS) Readout Stops Updating upon Returning from C3 State

Problem: Digital Thermal Sensor (DTS) Readout is provided in IA32_THERM_STATUS bits 22:16. Upon waking up from C3 low-power state, the DTS readout will no longer be updated.

Implication: Upon waking up from C3 low-power state, software cannot rely on DTS readout. Any thermal threshold interrupts that are enabled in IA32_THERM_INTERRUPT, will also be affected.

Workaround: It is possible for BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W59. LOCK# Asserted during a Special Cycle Shutdown Transaction May Unexpectedly Deassert

Problem: During a processor shutdown transaction, when LOCK# is asserted and if a DEFER# is received during a snoop phase and the Locked transaction is pipelined on the front side bus (FSB), LOCK# may unexpectedly deassert.

Implication: When this erratum occurs, the system may hang during shutdown. Intel has not observed this erratum with any commercially-available systems or software.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W60. Last Branch Records (LBR) Updates May Be Incorrect after a Task Switch

Problem: A Task-State Segment (TSS) task switch may incorrectly set the LBR_FROM value to the LBR_TO value.

Implication: The LBR_FROM will have the incorrect address of the Branch Instruction.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W61. Address Reported by Machine-Check Architecture (MCA) on Single-bit L2 ECC Errors May Be Incorrect

Problem: When correctable single-bit ECC errors occur in the L2 cache the address is logged in the MCA address register (MCi_ADDR). Under some scenarios, the address reported may be incorrect.

Implication: Software should not rely on the value reported in MCi_ADDR, for Single-bit L2 ECC errors.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W62. Disabling of Single-step On Branch Operation May Be Delayed following a POPFD Instruction

Problem: Disabling of Single-step On Branch Operation may be delayed, if the following conditions are met:

1. "Single Step On Branch Mode" is enabled (DebugCtlMSR.BTF and EFLAGS.TF are set)
2. POPFD used to clear EFLAGS.TF
3. A jump instruction (JMP, Jcc, etc.) is executed immediately after POPFD0.

Implication: Single-step On Branch mode may remain in effect for one instruction after the POPFD instruction disables it by clearing the EFLAGS.TF bit.

Workaround: There is no workaround for Single-Step operation in commercially-available software. The workaround for custom software is to execute at least one instruction following POPFD before issuing a JMP instruction.

Status: For the steppings affected, see the Summary Tables of Changes.

W63. Performance Monitoring Counters That Count External Bus Events May Report Incorrect Values after Processor Power State Transitions

Problem: Performance monitoring counters that count external bus events operate when the processor is in the Active state (C0). If a processor transitions to a new power state, these Performance monitoring counters will stop counting, even if the event being counted remains active.

Implication: After transitioning between processor power states, software may observe incorrect counts in Performance monitoring counters that count external bus events.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W64. VERW/VERR/LSL/LAR Instructions May Unexpectedly Update the Last Exception Record (LER) MSR

Problem: The LER MSR may be unexpectedly updated, if the resultant value of the Zero Flag (ZF) is zero after executing the following instructions:

1. VERR (ZF=0 indicates unsuccessful segment read verification)
2. VERW (ZF=0 indicates unsuccessful segment write verification)
3. LAR (ZF=0 indicates unsuccessful access rights load)
4. LSL (ZF=0 indicates unsuccessful segment limit load).

Implication: The value of the LER MSR may be inaccurate if VERW/VERR/LSL/LAR instructions are executed after the occurrence of an exception.

Workaround: Software exception handlers that rely on the LER MSR value should read the LER MSR before executing VERW/VERR/LSL/LAR instructions.

Status: For the steppings affected, see the Summary Tables of Changes.

W65. Performance Monitoring Events for Retired Floating Point Operations (C1h) May Not Be Accurate

Problem: Performance monitoring events that count retired floating point operations may be too high.

Implication: The Performance Monitoring Event may have an inaccurate count.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W66. Global Pages in the Data Translation Look-Aside Buffer (DTLB) May Not Be Flushed by RSM instruction before Restoring the Architectural State from SMRAM

Problem: The Resume from System Management Mode (RSM) instruction does not flush global pages from the Data Translation Look-Aside Buffer (DTLB) prior to reloading the saved architectural state.

Implication: If SMM turns on paging with global paging enabled and then maps any of linear addresses of SMRAM using global pages, RSM load may load data from the wrong location.

Workaround: Do not use global pages in system management mode.

Status: For the steppings affected, see the Summary Tables of Changes.

W67. Data Breakpoint/Single Step on MOV SS/POP SS May Be Lost after Entry into SMM

Problem: Data Breakpoint/Single Step exceptions are normally blocked for one instruction following MOV SS/POP SS instructions. Immediately after executing these instructions, if the processor enters SMM (System Management Mode), upon RSM (resume from SMM) operation, normal processing of Data Breakpoint/Single Step exceptions is restored.

Because of this erratum, Data Breakpoints/Single step exceptions on MOVSS/POPSS instructions may be lost under one of the following conditions.

1. Following SMM entry and after RSM, the next instruction to be executed is HLT or MWAIT
2. SMM entry after executing MOV SS/POP SS is the result of executing an I/O instruction that triggers a synchronous SMI (System Management Interrupt)

Implication: Data Breakpoints/Single step operation on MOV SS/POP SS instructions may be unreliable in the presence of SMIs.

Workaround: None Identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W68. Hardware Prefetch Performance Monitoring Events May Be Counted Inaccurately

Problem: Hardware prefetch activity is not accurately reflected in the hardware prefetch performance monitoring.

Implication: This erratum may cause inaccurate counting for all hardware prefetch performance monitoring events.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W69. Programming the Digital Thermal Sensor (DTS) Threshold May Cause Unexpected Thermal Interrupts

Problem: Software can enable DTS thermal interrupts by programming the thermal threshold and setting the respective thermal interrupt enable bit. When programming DTS value, the previous DTS threshold may be crossed. This will generate an unexpected thermal interrupt.

Implication: Software may observe an unexpected thermal interrupt occur after reprogramming the thermal threshold.

Workaround: In the ACPI/OS implement a workaround by temporarily disabling the DTS threshold interrupt before updating the DTS threshold value.

Status: For the steppings affected, see the Summary Tables of Changes.

W70. Removed – See Erratum W51

W71. Removed – See Erratum W50

W72. Removed – See Erratum W53

W73. SYSENTER/SYSEXIT Instructions Can Implicitly Load "Null Segment Selector" to SS and CS Registers

Problem: According to the processor specification, attempting to load a null segment selector into the CS and SS segment registers should generate a General Protection Fault (#GP). Although loading a null segment selector to the other segment registers is allowed, the processor will generate an exception when the segment register holding a null selector is used to access memory. However, the SYSENTER instruction can implicitly load a null value to the SS segment selector. This can occur if the value in SYSENTER_CS_MSR is between FFF8h and FFFBh when the SYSENTER instruction is executed. This behavior is part of the SYSENTER/SYSEXIT instruction definition; the content of the SYSTEM_CS_MSR is always incremented by 8 before it is loaded into the SS. This operation will set the null bit in the segment selector if a null result is generated, but it does not generate a #GP on the SYSENTER instruction itself. An exception will be generated as expected when the SS register is used to access memory, however. The SYSEXIT instruction will also exhibit this

behavior for both CS and SS when executed with the value in SYSENTER_CS_MSR between FFF0h and FFF3h, or between FFE8h and FFEBh, inclusive.

Implication: These instructions are intended for operating system use. If this erratum occurs (and the OS does not ensure that the processor never has a null segment selector in the SS or CS segment registers), the processor's behavior may become unpredictable, possibly resulting in system failure.

Workaround: Do not initialize the SYSTEM_CS_MSR with the values between FFF8h and FFFBh, FFF0h and FFF3h, or FFE8h and FFEBh before executing SYSENTER or SYSEXIT.

Status: For the steppings affected, see the Summary Tables of Changes.

W74. Using 2-M/4-M Pages When A20M# Is Asserted May Result in Incorrect Address Translations

Problem: An external A20M# pin if enabled forces address bit 20 to be masked (forced to zero) to emulates real-address mode address wraparound at 1 megabyte. However, if all of the following conditions are met, address bit 20 may not be masked.

- paging is enabled
- a linear address has bit 20 set
- the address references a large page
- A20M# is enabled

Implication: When A20M# is enabled and an address references a large page the resulting translated physical address may be incorrect. This erratum has not been observed with any commercially-available operating system.

Workaround: Operating systems should not allow A20M# to be enabled if the masking of address bit 20 could be applied to an address that references a large page. A20M# is normally only used with the first megabyte of memory.

Status: For the steppings affected, see the Summary Tables of Changes.

W75. CPUID Extended Feature Does Not Report Intel® Thermal Monitor 2 Support Correctly

Problem: Processors with no support for Intel® Thermal Monitor 2 will falsely report support for Intel Thermal Monitor 2 as enabled by setting TM2 (bit 8) in the Extended Feature Flag returned in ECX when executing CPUID with EAX=01H.

Implication: Extended Feature Flag TM2 cannot be used to identify processors where Intel Thermal Monitor 2 is disabled.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W76. Removed erratum due to redundancy

W77. Removed – See Erratum W32

W78. Premature Execution of a Load Operation Prior to Exception Handler Invocation

Problem: If any of the below circumstances occur it is possible that the load portion of the instruction will have executed before the exception handler is entered.

1. If an instruction that performs a memory load causes a code segment limit violation
2. If a waiting X87 floating-point instruction or MMX™ technology (MMX) instruction that performs a memory load has a floating-point exception pending
3. If an MMX or SSE/SSE2/SSE3/SSSE3 extensions (SSE) instruction that performs a memory load and has either CR0.EM=1 (Emulation bit set), or a floating-point Top-of-Stack (FP TOS) not equal to 0, or a DNA exception pending

Implication: In normal code execution where the target of the load operation is to write back memory there is no impact from the load being prematurely executed, nor from the restart and subsequent re-execution of that instruction by the exception handler. If the target of the load is to uncached memory that has a system side-effect, restarting the instruction may cause unexpected system behavior due to the repetition of the side-effect. Particularly, while CR0.TS [bit 3] is set, a MOVD/MOVQ with MMX/XMM register operands may issue a memory load before getting the DNA exception.

Workaround: Code which performs loads from memory that has side-effects can effectively workaround this behavior by using simple integer-based load instructions when accessing side-effect memory and by ensuring that all code is written such that a code segment limit violation cannot occur as a part of reading from side-effect memory.

Status: For the steppings affected, see the Summary Tables of Changes.

W79. Performance Monitoring Events for Retired Instructions (C0H) May Not Be Accurate

Problem: The INST_RETIRED performance monitor may miscount retired instructions as follows:

- Repeat string and repeat I/O operations are not counted when a hardware interrupt is received during or after the last iteration of the repeat flow.
- VMLAUNCH and VMRESUME instructions are not counted.
- HLT and MWAIT instructions are not counted. The following instructions, if executed during HLT or MWAIT events, are also not counted:
  - a) RSM from a C-state SMI during an MWAIT instruction.
  - b) RSM from an SMI during a HLT instruction.

Implication: There may be a smaller than expected value in the INST_RETIRED performance monitoring counter. The extent to which this value is smaller than expected is determined by the frequency of the above cases.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W80. #GP Fault Is NOT Generated on Writing IA32_MISC_ENABLE [34] When Execute Disable Bit Is Not Supported

Problem: #GP fault is not generated on writing to IA32_MISC_ENABLE [34] bit in a processor which does not support Execute Disable Bit functionality.

Implication: Writing to IA32_MISC_ENABLE [34] bit is silently ignored without generating a fault.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W81. This erratum has been removed. It does not apply to the Celeron M processor. Please see the *Summary Tables of Changes* for more information.

W82. Removed; See Erratum W26

W83. Writing Shared Unaligned Data That Crosses a Cache Line without Proper Semaphores or Barriers May Expose a Memory Ordering Issue

Problem: Software which is written so that multiple agents can modify the same shared unaligned memory location at the same time may experience a memory ordering issue if multiple loads access this shared data shortly thereafter. Exposure to this problem requires the use of a data write which spans a cache line boundary.

Implication: This erratum may cause loads to be observed out of order. Intel has not observed this erratum with any commercially-available software or system.

Workaround: Software should ensure at least one of the following is true when modifying shared data by multiple agents:

- The shared data is aligned
- Proper semaphores or barriers are used in order to prevent concurrent data accesses.

Status: For affected steppings see the Summary Tables of Changes.

W84. MSRs Actual Frequency Clock Count (IA32_APERF) or Maximum Frequency Clock Count (IA32_MPERF) May Contain Incorrect Data after a Machine Check Exception (MCE)

Problem: When an MCE occurs during execution of a RDMSR instruction for MSRs Actual Frequency Clock Count (IA32_APERF) or Maximum Frequency Clock Count (IA32_MPERF), the current and subsequent RDMSR instructions for these MSRs may contain incorrect data.

Implication: After an MCE event, accesses to the IA32_APERF and IA32_MPERF MSRs may return incorrect data. A subsequent reset will clear this condition.

Workaround: None identified.

Status: For affected steppings see the Summary Tables of Changes.

W85. Incorrect Address Computed for Last Byte of FXSAVE/FXRSTOR Image Leads to Partial Memory Update

Problem: A partial memory state save of the 512-byte FXSAVE image or a partial memory state restore of the FXRSTOR image may occur if a memory address exceeds the 64-kB limit while the processor is operating in 16-bit mode or if a memory address exceeds the 4-GB limit while the processor is operating in 32-bit mode.

Implication: FXSAVE/FXRSTOR will incur a #GP fault due to the memory limit violation as expected but the memory state may be only partially saved or restored.

Workaround: Software should avoid memory accesses that wrap around the respective 16-bit and 32-bit mode memory limits.

Status: For affected steppings see the Summary Tables of Changes.

W86. Values for LBR/BTS/BTM Will Be Incorrect after an Exit from SMM

Problem: After a return from SMM (System Management Mode), the CPU will incorrectly update the LBR (Last Branch Record) and the BTS (Branch Trace Store), hence rendering their data invalid. The corresponding data if sent out as a BTM on the system bus will also be incorrect. Note: This issue would only occur when one of the 3 above-mentioned debug support facilities are used.

Implication: The value of the LBR, BTS, and BTM immediately after an RSM operation should not be used.

Workaround: None identified.

Status: For affected steppings see the Summary Tables of Changes.

W87. Using Memory Type Aliasing with Memory Types WB/WT May Lead to Unpredictable Behavior

Problem: Memory type aliasing occurs when a single physical page is mapped to two or more different linear addresses, each with different memory type. Memory type aliasing with the memory types WB and WT may cause the processor to perform incorrect operations leading to unpredictable behavior.

Implication: Software that uses aliasing of WB and WT memory types may observe unpredictable behavior. Intel chipset-based platforms are not affected by this erratum.

Workaround: None identified. Intel does not support the use of WB and WT page memory type aliasing.

Status: For affected steppings see the Summary Tables of Changes.

W88. Performance Monitoring Event FP_ASSIST May Not Be Accurate

Problem: Performance monitoring event FP_ASSIST (11H) may be inaccurate as assist events will be counted twice per actual assist in the following specific cases:

- FADD and FMUL instructions with a Not a Number (NaN) operand and a memory operand.
- FDIV instruction with zero operand value in memory

In addition, an assist event may be counted when DAZ (Denormals-Are-Zeros) and FTZ (Flush-To-Zero) flags are turned on even though no actual assist occurs.

Implication: The counter value for performance monitoring event FP_ASSIST (11H) may be larger than expected. The size of the error is dependent on the number of occurrences of the above condition while the event is active.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W89. The BS Flag in DR6 May Be Set for Non-Single-Step #DB Exception

Problem: DR6 BS (Single Step, bit 14) flag may be incorrectly set when the TF (Trap Flag, bit 8) of the EFLAGS Register is set, and a #DB (Debug Exception) occurs due to one of the following:

- DR7 GD (General Detect, bit 13) being bit set;
- INT1 instruction;
- Code breakpoint

Implication: The BS flag may be incorrectly set for non-single-step #DB exception.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W90. BTM/BTS Branch-From Instruction Address May Be Incorrect for Software Interrupts

Problem: When BTM (Branch Trace Message) or BTS (Branch Trace Store) is enabled, a software interrupt may result in the overwriting of BTM/BTS branch-from instruction address by the LBR (Last Branch Record) branch-from instruction address.

Implication: A BTM/BTS branch-from instruction address may get corrupted for software interrupts.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W91. Erratum Removed

W92. Unaligned Accesses to Paging Structures May Cause the Processor to Hang

Problem: When an unaligned access is performed on paging structure entries, accessing a portion of two different entries simultaneously, the processor may live lock

Implication: When this erratum occurs, the processor may live lock causing a system hang.

Workaround: Do not perform unaligned accesses on paging structure entries.

Status: For affected steppings see the Summary Table of Changes.

W93. INVLPG Operation for Large (2M/4M) Pages May Be Incomplete under Certain Conditions

Problem: The INVLPG instruction may not completely invalidate Translation Look-aside Buffer (TLB) entries for large pages (2M/4M) when both of the following conditions exist:

- Address range of the page being invalidated spans several Memory Type Range Registers (MTRRs) with different memory types specified
- INVLPG operation is preceded by a Page Assist Event (Page Fault (#PF) or an access that results in either A or D bits being set in a Page Table Entry (PTE))

Implication: Stale translations may remain valid in TLB after a PTE update resulting in unpredictable system behavior. Intel has not observed this erratum with any commercially available software.

Workaround: Software should ensure that the memory type specified in the MTRRs is the same for the entire address range of the large page.

Status: For affected steppings see the Summary Table of Changes.

W94. Page Access Bit May Be Set Prior to Signaling a Code Segment Limit Fault

Problem: If code segment limit is set close to the end of a code page, then due to this erratum the memory page Access bit (A bit) may be set for the subsequent page prior to general protection fault on code segment limit.

Implication: When this erratum occurs, a non-accessed page which is present in memory and follows a page that contains the code segment limit may be tagged as accessed.

Workaround: Erratum can be avoided by placing a guard page (non-present or non-executable page) as the last page of the segment or after the page that includes the code segment limit

Status: For affected steppings see the Summary Table of Changes.

W95. Microcode Updates Performed During VMX Non-root Operation Could Result in Unexpected Behavior

Problem: When Intel® Virtualization Technology is enabled, microcode updates are allowed only during VMX root operations. Attempts to apply microcode updates while in VMX non-root operation should be silently ignored. Due to this erratum, the processor may allow microcode updates during VMX non-root operations if not explicitly prevented by the host software.

Implication: Microcode updates performed in non-root operation may result in unexpected system behavior.

Workaround: Host software should intercept and prevent loads to IA32_BIOS_UPDT_TRIG MSR (79H) during VMX non-root operations. There are two mechanism that can be used (1) Enabling MSR access protection in the VM-execution controls or (2) Enabling selective MSR protection of IA32_BIOS_UPDT_TRIG MSR.

Status: For affected steppings see the Summary Table of Changes.

W96. Single Step Interrupts with Floating Point Exception Pending May Be Mishandled

Problem: In certain circumstances, when a floating point exception (#MF) is pending during single-step execution, processing of the single-step debug exception (#DB) may be mishandled.

Implication: When this erratum occurs, #DB will be incorrectly handled as follows:

- #DB is signaled before the pending higher priority #MF (Interrupt 16)
- #DB is generated twice on the same instruction

Workaround: None Identified

Status: For the steppings affected, see the Summary Tables of Changes.

W97. Fault on ENTER Instruction May Result in Unexpected Values on Stack Frame

Problem: The ENTER instruction is used to create a procedure stack frame. Due to this erratum, if execution of the ENTER instruction results in a fault, the dynamic storage area of the resultant stack frame may contain unexpected values (i.e. residual stack data as a result of processing the fault).

Implication: Data in the created stack frame may be altered following a fault on the ENTER instruction. Please refer to "Procedure Calls For Block-Structured Languages" in the *Intel® 64 and IA-32 Architecture Software Developer's Manual*, *Vol. 1, Basic Architecture*, for information on the usage of the ENTER instructions. This erratum is not expected to occur in ring 3. Faults are usually processed in ring 0 and stack switch occurs when transferring to ring 0. Intel has not observed this erratum on any commercially available software.

Workaround: None Identified

Status: For the steppings affected, see the Summary Tables of Changes.

W98. SYSRET May Incorrectly Clear RF (Resume Flag) in the RFLAGS Register

Problem: In normal operation, SYSRET will restore the value of RFLAGS from R11 (the value previously saved upon execution of the SYSCALL instruction). Due to this erratum, the RFLAGS.RF bit will be unconditionally cleared after execution of the SYSRET instruction.

Implication: The SYSRET instruction can not be used if the RF flag needs to be set after returning from a system call. Intel has not observed this erratum with any commercially available software.

Workaround: Use the IRET instruction to return from a system call, if RF flag has to be set after the return.

Status: For the steppings affected, see the Summary Tables of Changes.

W99. General Protection Fault (#GP) for Instructions Greater Than 15 Bytes May Be Preempted

Problem: When the processor encounters an instruction that is greater than 15 bytes in length, a #GP is signaled when the instruction is decoded. Under some circumstances, the #GP fault may be preempted by another lower priority fault (for example, Page Fault (#PF)). However, if the preempting lower priority faults are resolved by the operating system and the instruction retried, a #GP fault will occur.

Implication: Software may observe a lower-priority fault occurring before or in lieu of a #GP fault. Instructions of greater than 15 bytes in length can only occur if redundant prefixes are placed before the instruction.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W100. CS Limit Violation on RSM May be Serviced before Higher Priority Interrupts/Exceptions

Problem: When the processor encounters a CS (Code Segment) limit violation, a #GP (General Protection Exception) fault is generated after all higher priority Interrupts and exceptions are serviced. Because of this erratum, if RSM (Resume from System Management Mode) returns to execution flow where a CS limit violation occurs, the #GP fault may be serviced before a higher priority Interrupt or Exception (for example NMI (Non-Maskable Interrupt), Debug break (#DB), Machine Check (#MC), etc.).

Implication: Operating systems may observe a #GP fault being serviced before higher priority Interrupts and Exceptions.

Workaround: None Identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W101. Count Value for Performance-Monitoring Counter PMH_PAGE_WALK May Be Incorrect

Problem: Performance-Monitoring Counter PMH_PAGE_WALK is used to count the number of page walks resulting from Data Translation Look-Aside Buffer (DTLB) and Instruction Translation Look-Aside (ITLB) misses. Under certain conditions, this counter may be incorrect.

Implication: There may be small errors in the accuracy of the counter.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W102. Performance Monitoring Event For Number of Reference Cycles When the Processor Is Not Halted (3CH) Does Not Count According to the Specification

Problem: The CPU_CLK_UNHALTED performance monitor with mask 1 counts bus clock cycles instead of counting the core clock cycles at the maximum possible ratio. The maximum possible ratio is computed by dividing the maximum possible core frequency by the bus frequency.

Implication: The CPU_CLK_UNHALTED performance monitor with mask 1 counts a value lower than expected. The value is lower by exactly one multiple of the maximum possible ratio.

Workaround: Multiply the performance monitor value by the maximum possible ratio.

Status: For the steppings affected, see the Summary Tables of Changes.

W103. Sequential Code Fetch to Non-canonical Address May Have Nondeterministic Results

Problem: If code sequentially executes off the end of the positive canonical address space (falling through from address 00007fffffffffff to non- canonical address 0000800000000000), under some circumstances the code fetch will be converted to a canonical fetch at address ffff800000000000.

Implication: Due to this erratum, the processor may transfer control to an unintended address. The result of fetching code at that address is unpredictable and may include an unexpected trap or fault, or execution of the instructions found there.

Workaround: If the last page of the positive canonical address space is not allocated for code (4K page at 00007ffffffff000 or 2M page at 00007fffffe00000) then the problem cannot occur.

Status: For the steppings affected, see the Summary Tables of Changes.

W104. Power Cycle Generator Bus Transactions Will Not Be Counted As Local Transactions

Problem: The following Bus Transaction Performance Monitor events are supposed to count all local transactions:

- BUS_TRANS_ IO (Event: 6Ch)
- BUS_TRANS_ANY (Event: 70h)

Due to this erratum when the BUS_TRANS_IO and BUS_TRANS_ANY events with Agent Specificity clear (Umask bit [13] =0), Power Cycle Generator events such as hardware coordination IORead and StopGrant are not counted as local transactions.

Implication: Power cycle generator bus transactions may not be counted as local transactions when bus transaction performance monitor events are enabled.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W105. EIP May Be Incorrect after Shutdown in IA-32e Mode

Problem: When the processor is going into shutdown state the upper 32 bits of the instruction pointer may be incorrect. This may be observed if the processor is taken out of shutdown state by NMI#.

Implication: A processor that has been taken out of the shutdown state may have an incorrect EIP. The only software which would be affected is diagnostic software that relies on a valid EIP.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W106. (E)CX May Get Incorrectly Updated When Performing Fast String REP MOVS or Fast String REP STOS with Large Data Structures

Problem: When performing Fast String REP MOVS or REP STOS commands with data structures [(E)CX*Data Size] larger than the supported address size structure (64K for 16-bit address size and 4G for 32-bit address size) some addresses may be processed more than once. After an amount of data greater than or equal to the address size structure has been processed, external events (such as interrupts) will cause the (E)CX registers to be increment by a value that corresponds to 64K bytes for 16 bit address size and 4G bytes for 32 bit address size.

Implication: (E)CX may contain an incorrect count which may cause some of the MOVS or STOS operations to re-execute. Intel has not observed this erratum with any commercially available software.

Workaround: Do not use values in (E)CX that when multiplied by the data size give values larger than the address space size (64 kB for 16-bit address size and 4 GB for 32-bit address size).

Status: For the steppings affected, see the Summary Tables of Changes.

W107. Upper 32 bits of 'From' Address Reported through BTMs or BTSs May Be Incorrect

Problem: When a far transfer switches the processor from 32-bit mode to IA-32e mode, the upper 32 bits of the 'From' (source) addresses reported through the BTMs (Branch Trace Messages) or BTSs (Branch Trace Stores) may be incorrect.

Implication: The upper 32 bits of the 'From' address debug information reported through BTMs or BTSs may be incorrect during this transition.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W108. Unsynchronized Cross-Modifying Code Operations Can Cause Unexpected Instruction Execution Results

Problem: The act of one processor, or system bus master, writing data into a currently executing code segment of a second processor with the intent of having the second processor execute that data as code is called cross-modifying code (XMC). XMC that does not force the second processor to execute a synchronizing instruction, prior to execution of the new code, is called unsynchronized XMC. Software using unsynchronized XMC to modify the instruction byte stream of a processor can see unexpected or unpredictable execution behavior from the processor that is executing the modified code.

Implication: In this case, the phrase "unexpected or unpredictable execution behavior" encompasses the generation of most of the exceptions listed in the *Intel® 64 and IA-32 Architecture Software Developer's Manual Volume 3: System Programming Guide*, including a General Protection

Fault (GPF) or other unexpected behaviors. In the event that unpredictable execution causes a GPF the application executing the unsynchronized XMC operation would be terminated by the operating system.

Workaround: In order to avoid this erratum, programmers should use the XMC synchronization algorithm as detailed in the *Intel® 64 and IA-32 Architecture Software Developer's Manual, Volume 3: System Programming Guide*, Section: *Handling Self- and Cross-Modifying Code*.

Status: For the steppings affected, see the Summary Tables of Changes.

W109. Split Locked Stores May Not Trigger the Monitoring Hardware

Problem: Logical processors normally resume program execution following the MWAIT, when another logical processor performs a write access to a WB cacheable address within the address range used to perform the MONITOR operation. Due to this erratum, a logical processor may not resume execution until the next targeted interrupt event or O/S timer tick following a locked store that spans across cache lines within the monitored address range.

Implication: The logical processor that executed the MWAIT instruction may not resume execution until the next targeted interrupt event or O/S timer tick in the case where the monitored address is written by a locked store which is split across cache lines.

Workaround: Do not use locked stores that span cache lines in the monitored address range.

Status: For the steppings affected, see the Summary Tables of Changes.

W110. REP CMPS/SCAS Operations May Terminate Early in 64-bit Mode when RCX >= 0X100000000

Problem: REP CMPS (Compare String) and SCAS (Scan String) instructions in 64-bit mode may terminate before the count in RCX reaches zero if the initial value of RCX is greater than or equal to 0X100000000.

Implication: Early termination of REP CMPS/SCAS operation may be observed and RFLAGS may be incorrectly updated.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W111. FXSAVE/FXRSTOR Instructions which Store to the End of the Segment and Cause a Wrap to a Misaligned Base Address (Alignment <= 0x10h) May Cause FPU Instruction or Operand Pointer Corruption

Problem: If a FXSAVE/FXRSTOR instruction stores to the end of the segment causing a wrap to a misaligned base address (alignment <= 0x10h), and one of the following conditions is satisfied:

1. 32-bit addressing, obtained by using address-size override, when in 64-bit mode.

2. 16-bit addressing in legacy or compatibility mode.

Then, depending on the wrap-around point, one of the below saved values may be corrupted:

- FPU Instruction Pointer Offset
- FPU Instruction Pointer Selector
- FPU Operand Pointer Selector
- FPU Operand Pointer Offset

Implication: This erratum could cause FPU instruction or operand pointer corruption and may lead to unexpected operations in the floating point exception handler.

Workaround: Avoid segment base misalignment and address wrap-around at the segment boundary.

Status: For the steppings affected, see the Summary Tables of Changes.

W112. PREFETCHh Instruction Execution under Some Conditions May Lead to Processor Livelock

Problem: PREFETCHh instruction execution after a split load and dependent upon ongoing store operations may lead to processor livelock.

Implication: Due to this erratum, the processor may livelock.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W113. PREFETCHh Instructions May Not Be Executed When Alignment Check (AC) is Enabled

Problem: PREFETCHT0, PREFETCHT1, PREFETCHT2 and PREFETCHNTA instructions may not be executed when alignment check is enabled.

Implication: PREFETCHh instructions may not perform the data prefetch if Alignment Check is enabled.

Workaround: Clear the AC flag (bit 18) in the EFLAGS register and/or the AM bit (bit 18) of Control Register CR0 to disable alignment checking.

Status: For the steppings affected, see the Summary Tables of Changes.

W114. Upper 32 Bits of the FPU Data (Operand) Pointer in the FXSAVE Memory Image May Be Unexpectedly All 1's after FXSAVE

Problem: The upper 32 bits of the FPU Data (Operand) Pointer may incorrectly be set to all 1's instead of the expected value of all 0's in the FXSAVE memory image if all of the following conditions are true:

- The processor is in 64-bit mode.

- The last floating point operation was in compatibility mode
- Bit 31 of the FPU Data (Operand) Pointer is set.
- An FXSAVE instruction is executed

Implication: Software depending on the full FPU Data (Operand) Pointer may behave unpredictably.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W115. Performance Monitor IDLE_DURING_DIV (18h) Count May Not Be Accurate

Problem: Performance monitoring events that count the number of cycles the divider is busy and no other execution unit operation or load operation is in progress may not be accurate.

Implication: The counter may reflect a value higher or lower than the actual number of events.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W116. SYSCALL Immediately after Changing EFLAGS.TF May Not Behave According to the New EFLAGS.TF

Problem: If a SYSCALL instruction follows immediately after EFLAGS.TF was updated and IA32_FMASK.TF (bit 8) is cleared, then under certain circumstances SYSCALL may behave according to the previous EFLAGS.TF.

Implication: When the problem occurs, SYSCALL may generate an unexpected debug exception, or may skip an expected debug exception.

Workaround: Mask EFLAGS.TF by setting IA32_FMASK.TF (bit 8).

Status: For the steppings affected, see the Summary Tables of Changes.

W117. IA32_FMASK Is Reset during an INIT

Problem: IA32_FMASK MSR (0xC0000084) is reset during INIT.

Implication: If an INIT takes place after IA32_FMASK is programmed, the processor will overwrite the value back to the default value.

Workaround: Operating system software should initialize IA32_FMASK after INIT.

Status: For the steppings affected, see the Summary Tables of Changes.

W118. IO_SMI Indication in SMRAM State Save Area May Be Set Incorrectly

Problem: The IO_SMI bit in SMRAM's location 7FA4H is set to "1" by the CPU to indicate a System Management Interrupt (SMI) occurred as the result of executing an instruction that reads from an I/O port. Due to this erratum, the IO_SMI bit may be incorrectly set by:

- A non-I/O instruction.
- SMI is pending while a lower priority event interrupts.
- A REP I/O read.
- An I/O read that redirects to MWAIT.

Implication: SMM handlers may get false IO_SMI indication.

Workaround: The SMM handler has to evaluate the saved context to determine if the SMI was triggered by an instruction that read from an I/O port. The SMM handler must not restart an I/O instruction if the platform has not been configured to generate a synchronous SMI for the recorded I/O port address.

Status: For the steppings affected, see the Summary Tables of Changes.

W119. LBR, BTS, BTM May Report a Wrong Address When an Exception / Interrupt Occurs in 64-bit Mode

Problem: An exception/interrupt event should be transparent to the LBR (Last Branch Record), BTS (Branch Trace Store) and BTM (Branch Trace Message) mechanisms. However, during a specific boundary condition where the exception/interrupt occurs right after the execution of an instruction at the lower canonical boundary (0x00007FFFFFFFFFFF) in 64-bit mode, the LBR return registers will save a wrong return address with bits 63 to 48 incorrectly sign extended to all 1's. Subsequent BTS and BTM operations which report the LBR will also be incorrect.

Implication: LBR, BTS and BTM may report incorrect information in the event of an exception/interrupt.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W120. A Thermal Interrupt Is Not Generated When the Current Temperature Is Invalid

Problem: When the DTS (Digital Thermal Sensor) crosses one of its programmed thresholds it generates an interrupt and logs the event (IA32_THERM_STATUS MSR (019Ch) bits [9, 7]). Due to this erratum, if the DTS reaches an invalid temperature (as indicated IA32_THERM_STATUS MSR bit[31]) it does not generate an interrupt even if one of the programmed thresholds is crossed and the corresponding log bits become set.

Implication: When the temperature reaches an invalid temperature the CPU does not generate a Thermal interrupt even if a programmed threshold is crossed.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W121. CMPSB, LODSB, or SCASB in 64-bit Mode with Count Greater or Equal to 2^{48} May Terminate Early

Problem: In 64-bit Mode CMPSB, LODSB, or SCASB executed with a repeat prefix and count greater than or equal to 2^{48} may terminate early. Early termination may result in one of the following.

- The last iteration not being executed
- Signaling of a canonical limit fault (#GP) on the last iteration

Implication: While in 64-bit mode, with count greater or equal to 2^{48}, repeat string operations CMPSB, LODSB or SCASB may terminate without completing the last iteration. Intel has not observed this erratum with any commercially available software.

Workaround: Do not use repeated string operations with RCX greater than or equal to 2^{48}.

Status: For the steppings affected, see the Summary Tables of Changes.

W122. Returning to Real Mode from SMM with EFLAGS.VM Set May Result in Unpredictable System Behavior

Problem: Returning back from SMM mode into real mode while EFLAGS.VM is set in SMRAM may result in unpredictable system behavior.

Implication: If SMM software changes the values of the EFLAGS.VM in SMRAM, it may result in unpredictable system behavior. Intel has not observed this behavior in commercially available software.

Workaround: SMM software should not change the value of EFLAGS.VM in SMRAM.

Status: For the steppings affected, see the Summary Tables of Changes.

W123. IRET under Certain Conditions May Cause an Unexpected Alignment Check Exception

Problem: In IA-32e mode, it is possible to get an Alignment Check Exception (#AC) on the IRET instruction even though alignment checks were disabled at the start of the IRET. This can only occur if the IRET instruction is returning from CPL3 code to CPL3 code. IRETs from CPL0/1/2 are not affected. This erratum can occur if the EFLAGS value on the stack has the AC flag set, and the interrupt handler's stack is misaligned. In IA-32e mode, RSP is aligned to a 16-byte boundary before pushing the stack frame.

Implication: In IA-32e mode, under the conditions given above, an IRET can get a #AC even if alignment checks are disabled at the start of the IRET. This erratum can only be observed with a software generated stack frame.

Workaround: Software should not generate misaligned stack frames for use with IRET.

Status: For the steppings affected, see the Summary Tables of Changes.

W124. Performance Monitoring Event FP_ASSIST May Not Be Accurate

Problem: Performance monitoring event FP_ASSIST (11H) may be inaccurate as assist events will be counted twice per actual assist in the following specific cases:

- FADD and FMUL instructions with a NaN(Not a Number) operand and a memory operand
- FDIV instruction with zero operand value in memory

In addition, an assist event may be counted when DAZ (Denormals-Are-Zeros) and FTZ (Flush-To-Zero) flags are turned on even though no actual assist occurs.

Implication: The counter value for the performance monitoring event FP_ASSIST (11H) may be larger than expected. The size of the error is dependent on the number of occurrences of the above conditions while the event is active.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W125. CPL-Qualified BTS May Report Incorrect Branch-From Instruction Address

Problem: CPL (Current Privilege Level)-qualified BTS (Branch Trace Store) may report incorrect branch-from instruction address under the following conditions:

- Either BTS_OFF_OS [9] or BTS_OFF_USR [10] is selected in IA32_DEBUGCTLC MSR (1D9H).
- Privilege-level transitions occur between CPL > 0 and CPL 0 or vice versa.

Implication: Due to this erratum, the from address reported by BTS may be incorrect for the described conditions.

Workaround: None identified

Status: For the steppings affected, see the Summary Tables of Changes.

W126. PEBS Does Not Always Differentiate between CPL-Qualified Events

Problem: Performance monitoring counter configured to sample PEBS (Precise Event Based Sampling) events at a certain privilege level may count samples at the wrong privilege level.

Implication: Performance monitoring counter may be higher than expected for CPL-qualified events. Do not use performance monitoring counters for precise event sampling when the precise event is dependent on the CPL value.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W127. PMI May Be Delayed to Next PEBS Event

Problem: After a PEBS (Precise Event-Based Sampling) event, the PEBS index is compared with the PEBS threshold, and the index is incremented with every event. If PEBS index is equal to the PEBS threshold, a PMI (Performance Monitoring Interrupt) should be issued. Due to this erratum, the PMI may be delayed by one PEBS event.

Implication: Debug Store Interrupt Service Routines may observe delay of PMI occurrence by one PEBS event.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W128. An Asynchronous MCE during a Far Transfer May Corrupt ESP

Problem: If an asynchronous machine check occurs during an interrupt, call through gate, FAR RET or IRET and in the presence of certain internal conditions, ESP may be corrupted.

Implication: If the MCE (Machine Check Exception) handler is called without a stack switch, then a triple fault will occur due to the corrupted stack pointer, resulting in a processor shutdown. If the MCE is called with a stack switch, for example when the CPL (Current Privilege Level) was changed or when going through an interrupt task gate, then the corrupted ESP will be saved on the stack or in the TSS (Task State Segment), and will not be used.

Workaround: Use an interrupt task gate for the machine check handler.

Status: For the steppings affected, see the Summary Tables of Changes.

W129. B0-B3 Bits in DR6 May Not Be Properly Cleared after Code Breakpoint

Problem: B0-B3 bits (breakpoint conditions detect flags, bits [3:0]) in DR6 may not be properly cleared when the following sequence happens:

1. POP instruction to SS (Stack Segment) selector.

2. Next instruction is FP (Floating Point) that gets FP assist followed by code breakpoint.

Implication: B0-B3 bits in DR6 may not be properly cleared.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W130. REP Store Instructions in a Specific Situation May Cause the Processor to Hang

Problem: During a series of REP (repeat) store instructions a store may try to dispatch to memory prior to the actual completion of the instruction. This behavior depends on the execution order of the instructions, the timing of a speculative jump and the timing of an uncacheable memory store. All types of REP store instructions are affected by this erratum.

Implication: When this erratum occurs, the processor may live lock and/or result in a system hang.

Workaround: It is possible for BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W131. Performance Monitor SSE Retired Instructions May Return Incorrect Values

Problem: The SIMD_INST_RETIRED (Event: C7H) is used to track retired SSE instructions. Due to this erratum, the processor may inaccurately count certain types of instructions resulting in values higher than the number of actual retired SSE instructions.

Implication: The event monitor instruction SIMD_INST_RETIRED may report count higher than expected.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W132. Performance Monitoring Events for L1 and L2 Miss May Not Be Accurate

Problem: Performance monitoring events 0CBh with an event mask value of 02h or 08h (MEM_LOAD_RETIRED.L1_LINE_MISS or MEM_LOAD_RETIRED.L2_LINE_MISS) may under count the cache miss events.

Implication: These performance monitoring events may show a count which is lower than expected; the amount by which the count is lower is dependent on other conditions occurring on the same load that missed the cache.

Workaround: None Identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W133. A MOV Instruction from CR8 Register with 16 Bit Operand Size Will Leave Bits 63:16 of the Destination Register Unmodified

Problem: Moves to/from control registers are supposed to ignore REW.W and the 66H (operand size) prefix. In systems supporting Intel Virtualization Technology, when the processor is operating in VMX non-root operation and "use TPR shadow" VM-execution control is set to 1, a MOV instruction from CR8 with a 16 bit operand size (REX.W =0 and

66H prefix) will only store 16 bits and leave bits 63:16 at the destination register unmodified, instead of storing zeros in them.

Implication: Intel has not observed this erratum with any commercially available software.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W134. Debug Register May Contain Incorrect Information on a MOVSS or POPSS Instruction followed by SYSRET

Problem: In IA-32e mode, if a MOVSS or POPSS instruction with a debug breakpoint is followed by the SYSRET instruction; incorrect information may exist in the Debug Status Register (DR6).

Implication: When debugging or when developing debuggers, this behavior should be noted. This erratum does not occur under normal usage of the MOVSS or POPSS instructions (that is, following them with a MOV ESP instruction).

Workaround: Do not attempt to put a breakpoint on MOVSS and POPSS instructions that are followed by a SYSRET.

Status: For the steppings affected, see the Summary Tables of Changes.

W135. Update of Read/Write (R/W) or User/Supervisor (U/S) or Present (P) Bits without TLB Shootdown May Cause Unexpected Processor Behavior

Problem: Updating a page table entry by changing R/W, U/S or P bits without TLB shootdown (as defined by the 4 step procedure in "Propagation of Page Table and Page Directory Entry Changes to Multiple Processors" In volume 3A of the *Intel® 64 and IA-32 Architecture Software Developer's Manual*), in conjunction with a complex sequence of internal processor micro-architectural events, may lead to unexpected processor behavior.

Implication: This erratum may lead to livelock, shutdown or other unexpected processor behavior. Intel has not observed this erratum with any commercially available system.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W136. Update of Attribute Bits on Page Directories without Immediate TLB Shootdown May Cause Unexpected Processor Behavior

Problem: Updating a page directory entry (or page map level 4 table entry or page directory pointer table entry in IA-32e mode) by changing R/W, U/S or P bits without immediate TLB shootdown (as described by the 4 step

procedure in "Propagation of Page Table and Page Directory Entry Changes to Multiple Processors" In Volume 3A of the *Intel® 64 and IA-32 Architecture Software Developer's Manual*), in conjunction with a complex sequence of internal processor micro-architectural events, may lead to unexpected processor behavior.

Implication: This erratum may lead to livelock, shutdown or other unexpected processor behavior. Intel has not observed this erratum with any commercially available system.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W137. Invalid Instructions May Lead to Unexpected Behavior

Problem: Invalid instructions due to undefined opcodes or instructions exceeding the maximum instruction length (due to redundant prefixes placed before the instruction) may lead, under complex circumstances, to unexpected behavior.

Implication: The processor may behave unexpectedly due to invalid instructions. Intel has not observed this erratum with any commercially available software.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W138. EFLAGS, CR0, CR4 and the EXF4 Signal May Be Incorrect after Shutdown

Problem: When the processor is going into shutdown due to an RSM inconsistency failure, EFLAGS, CR0 and CR4 may be incorrect. In addition the EXF4 signal may still be asserted. This may be observed if the processor is taken out of shutdown by NMI#.

Implication: A processor that has been taken out of shutdown may have an incorrect EFLAGS, CR0 and CR4. In addition the EXF4 signal may still be asserted.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W139. Performance Monitoring Counter MACRO_INSTS.DECODED May Not Count Some Decoded Instructions

Problem: MACRO_INSTS.DECODED performance monitoring counter (Event 0AAH, Umask 01H) counts the number of macro instructions decoded, but not necessarily retired. The event is undercounted when the decoded instructions are a complete loop iteration that is decoded in one cycle and the loop is streamed by the LSD (Loop Stream Detector), as

described in the Optimizing the Front End section of the *Intel® 64 and IA-32 Architecture Software Developer's Manual*

Implication: The count value returned by the performance monitoring counter MACRO_INST.DECODED may be lower than expected. The degree of undercounting is dependent on the occurrence of loop iterations that are decoded in one cycle and whether the loop is streamed by the LSD while the counter is active.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W140. The Stack May be Incorrect as a Result of VIP/VIF Check on SYSEXIT and SYSRET

Problem: The stack size may be incorrect under the following scenario:

1. The stack size was changed due to a SYSEXIT or SYSRET
2. PVI (Protected Mode Virtual Interrupts) mode was enabled (CR4.PVI == 1)
3. Both the VIF (Virtual Interrupt Flag) and VIP (Virtual Interrupt Pending) flags of the EFLAGS register are set.

Implication: If this erratum occurs the stack size may be incorrect, consequently this result in unpredictable system behavior. Intel has not observed this erratum with any commercially available software.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W141. Performance Monitoring Event SIMD_UOP_TYPE_EXEC.MUL Is Counted Incorrectly for PMULUDQ Instruction

Problem: Performance Monitoring Event SIMD_UOP_TYPE_EXEC.MUL (Event select 0B3H, Umask 01H) counts the number of SIMD packed multiply micro-ops executed. The count for PMULUDQ micro-ops might be lower than expected. No other instruction is affected.

Implication: The count value returned by the performance monitoring event SIMD_UOP_TYPE_EXEC.MUL may be lower than expected. The degree of undercount depends on actual occurrences of PMULUDQ instructions, while the counter is active.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W142. Storage of PEBS Record Delayed Following Execution of MOV SS or STI

Problem: When a performance monitoring counter is configured for PEBS (Precise Event Based Sampling), overflow of the counter results in storage of a PEBS record in the PEBS buffer. The information in the PEBS record

represents the state of the next instruction to be executed following the counter overflow. Due to this erratum, if the counter overflow occurs after execution of either MOV SS or STI, storage of the PEBS record is delayed by one instruction.

Implication: When this erratum occurs, software may observe storage of the PEBS record being delayed by one instruction following execution of MOV SS or STI. The state information in the PEBS record will also reflect the one instruction delay.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W143. Store Ordering May be Incorrect between WC and WP Memory Types

Problem: According to *Intel® 64 and IA-32 Architecture Software Developer's Manual, Volume 3A, Methods of Caching Available*, WP (Write Protected) stores should drain the WC (Write Combining) buffers in the same way as UC (Uncacheable) memory type stores do. Due to this erratum, WP stores may not drain the WC buffers.

Implication: Memory ordering may be violated between WC and WP stores.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W144. Updating Code Page Directory Attributes without TLB Invalidation May Result in Improper Handling of Code #PF

Problem: Code #PF (Page Fault exception) is normally handled in lower priority order relative to both code #DB (Debug Exception) and code Segment Limit Violation #GP (General Protection Fault). Due to this erratum, code #PF may be handled incorrectly, if all of the following conditions are met:

- A PDE (Page Directory Entry) is modified without invalidating the corresponding TLB (Translation Look-aside Buffer) entry
- Code execution transitions to a different code page such that both
  - The target linear address corresponds to the modified PDE
  - The PTE (Page Table Entry) for the target linear address has an A (Accessed) bit that is clear
- One of the following simultaneous exception conditions is present following the code transition
  - Code #DB and code #PF
  - Code Segment Limit Violation #GP and code #PF

Implication: Software may observe either incorrect processing of code #PF before code Segment Limit Violation #GP or processing of code #PF in lieu of code #DB.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W145. Performance Monitoring Event CPU_CLK_UNHALTED.REF May Not Count Clock Cycles According to the Processors Operating Frequency

Problem: Performance Counter MSR_PERF_FIXED_CTR2 (MSR 30BH) that counts CPU_CLK_UNHALTED.REF clocks, should count these clock cycles at a constant rate that is determined by the maximum resolved boot frequency, as programmed by BIOS. Due to this erratum, the rate is instead, set by the maximum core-clock to bus-clock ratio of the processor, as indicated by hardware.

Implication: No functional impact as a result of this erratum. If the maximum resolved boot frequency as programmed by BIOS is different from the frequency implied by the maximum core-clock to bus-clock ratio of the processor as indicated by hardware, then the following effects may be observed.

Workaround: Performance Monitoring Event CPU_CLK_UNHALTED.REF will count at a rate different than the TSC (Time Stamp Counter).

When running a system with several processors that have different maximum core-clock to bus-clock ratios, CPU_CLK_UNHALTED.REF monitoring events at each processor will be counted at different rates and therefore will not be comparable.

Calculate the ratio of the rates at which the TSC and the CPU_CLK_UNHALTED.REF performance monitoring event count (this can be done by measuring simultaneously their counted value while executing code) and adjust the CPU_CLK_UNHALTED.REF event count to the maximum resolved boot frequency using this ratio.

Status: For the steppings affected, see the Summary Tables of Changes.

W146. Performance Monitoring Event BR_INST_RETIRED May Count CPUID Instructions as Branches

Problem: Performance monitoring event BR_INST_RETIRED (C4H) counts retired branch instructions. Due to this erratum, two of its sub-events mistakenly count for CPUID instructions as well. Those sub events are: BR_INST_RETIRED.PRED_NOT_TAKEN (Umask 01H) and BR_INST_RETIRED.ANY (Umask 00H).

Implication: The count value returned by the performance monitoring event BR_INST_RETIRED.PRED_NOT_TAKEN or BR_INST_RETIRED.ANY may be higher than expected. The extent of over counting depends on the occurrence of CPUID instructions, while the counter is active.

Workaround: None Identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W147. W147. Performance Monitoring Event MISALIGN_MEM_REF May Over Count

Problem: Performance monitoring event MISALIGN_MEM_REF (05H) is used to count the number of memory accesses that cross an 8-byte boundary and are blocked until retirement. Due to this erratum, the performance monitoring event MISALIGN_MEM_REF also counts other memory accesses.

Implication: The performance monitoring event MISALIGN_MEM_REF may over count. The extent of over counting depends on the number of memory accesses retiring while the counter is active.

Workaround: None Identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W148. W148. A REP STOS/MOVS to a MONITOR/MWAIT Address Range May Prevent Triggering of the Monitoring Hardware

Problem: The MONITOR instruction is used to arm the address monitoring hardware for the subsequent MWAIT instruction. The hardware is triggered on subsequent memory store operations to the monitored address range. Due to this erratum, REP STOS/MOVS fast string operations to the monitored address range may prevent the actual triggering store to be propagated to the monitoring hardware.

Implication: A logical processor executing an MWAIT instruction may prevent program execution if a REP STOS/MOVS targets the monitored address range.

Workaround: It is possible for BIOS to contain a workaround for this erratum. Optionally, in order to have program execution continue immediately, REP STOS/MOVS should be prevented from targeting the same cache line as the monitored address range.

Status: For the steppings affected, see the Summary Tables of Changes.

W149. W149. False Level One Data Cache Parity Machine-Check Exceptions May Be Signaled

Problem: Executing an instruction stream containing invalid instructions/data may generate a false Level One Data Cache parity machine-check exception.

Implication: The false Level One Data Cache parity machine-check exception is reported as an uncorrected machine-check error. An uncorrected machine-check error is treated as a fatal exception by the operating system and may cause a shutdown and/or reboot.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W150. W150. A Memory Access May Get a Wrong Memory Type following a #GP due to WRMSR to an MTRR Mask

Problem: The TLB (Translation Lookaside Buffer) may indicate a wrong memory type on a memory access to a large page (2-M/4-M Byte) following the recovery from a #GP (General Protection Fault) due to a WRMSR to one of the IA32_MTRR_PHYSMASKn MSRs with reserved bits set.

Implication: When this erratum occurs, a memory access may get an incorrect memory type leading to unexpected system operation. As an example, an access to a memory mapped I/O device may be incorrectly marked as cacheable, become cached, and never make it to the I/O device. Intel has not observed this erratum with any commercially available software.

Workaround: Software should not attempt to set reserved bits of IA32_MTRR_PHYSMASKn MSRs.

Status: For the steppings affected, see the Summary Tables of Changes.

W151. W151. PMI While LBR Freeze Enabled May Result in Old/Out-of-date LBR Information.

Problem: When Precise Event-Based Sampling (PEBS) is configured with Performance Monitoring Interrupt (PMI) on PEBS buffer overflow enabled and Last Branch Record (LBR) Freeze on PMI enabled by setting FREEZE_LBRS_ON_PMI flag (bit 11) to 1 in IA32_DEBUGCTL (MSR 1D9H), the LBR stack is frozen upon the occurrence of a hardware PMI request. Due to this erratum, the LBR freeze may occur too soon (i.e. before the hardware PMI request).

Implication: Following a PMI occurrence, the PMI handler may observe old/out-of-date LBR information that does not describe the last few branches before the PEBS sample that triggered the PMI.

Workaround: None Identified

Status: For the steppings affected, see the Summary Tables of Changes.

W152. Performance Monitoring Event FP_MMX_TRANS_TO_MMX May Not Count Some Transitions

Problem: Performance Monitor Event FP_MMX_TRANS_TO_MMX (Event CCH, Umask 01H) counts transitions from x87 Floating Point (FP) to MMX™ instructions. Due to this erratum, if only a small number of MMX instructions (including EMMS) are executed immediately after the last FP instruction, a FP to MMX transition may not be counted.

Implication: The count value for Performance Monitoring Event FP_MMX_TRANS_TO_MMX may be lower than expected. The degree of undercounting is dependent on the occurrences of the erratum condition while the counter is active. Intel has not observed this erratum with any commercially available software.

Workaround: None Identified.

Status: For the steppings affected, see the Summary Tables of Changes.

W153. A WB Store Following a REP STOS/MOVS of FXSAVE May Lead to Memory-Ordering Violations

Problem: Under certain conditions, as described in the Software Developers Manual section "Out-of-Order Stores For String Operations in Pentium 4, Intel Xeon, and P6 Family Processors", the processor may perform REP MOVS or REP STOS as write combining stores (referred to as "fast strings") for optimal performance. FXSAVE may also be internally implemented using write combining stores. Due to this erratum, stores of a WB (write back) memory type to a cache line previously written by a preceding fast string/FXSAVE instruction may be observed before string/FXSAVE stores.

Implication: A write-back store may be observed before a previous string or FXSAVE related store. Intel has not observed this erratum with any commercially available software.

Workaround: None Identified.

Status: For the steppings affected, see the Summary Tables of Changes. §

W154. RSM Instruction Execution under Certain Conditions May Cause Processor Hang or Unexpected Instruction Execution

Problem: RSM instruction execution, under certain conditions triggered by a complex sequence of internal processor micro-architectural events, may lead to processor hang, or unexpected instruction execution results.

Implication: In the above sequence, the processor may live lock or hang, or RSM instruction may restart the interrupted processor context through a nondeterministic EIP offset in the code segment, resulting in unexpected instruction execution, unexpected exceptions or system hang. Intel has not observed this erratum with any commercially available software.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary Tables of Changes.

W152. W155: Benign Exception after a Double Fault May Not Cause a Triple Fault Shutdown

Problem: According to the *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3A, "Exception and Interrupt Reference"*, if another exception occurs while attempting to call the double-fault handler, the processor enters shutdown mode. However due to this erratum, only Contributory Exceptions and Page Faults will cause a triple fault shutdown, whereas a benign exception may not.

Implication: If a benign exception occurs while attempting to call the double-fault handler, the processor may hang or may handle the benign exception. Intel has not observed this erratum with any commercially available software.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

Specification Changes

There are no Specification Changes in this Specification Update revision.

§

Specification Clarifications

W1. Removed

See the Revision History for details.

W2. Removed

See the Revision History for details.

W3. Clarification of TRANSLATION LOOKASIDE BUFFERS (TLBS) Invalidation

Section 10.9 INVALIDATING THE TRANSLATION LOOKASIDE BUFFERS (TLBS) of the *Intel® 64 and IA-32 Architecture Software Developer's Manual, Volume 3A: System Programming Guide* will be modified to include the presence of page table structure caches, such as the page directory cache, which Intel processors implement. This information is needed to aid operating systems in managing page table structure invalidations properly.

Intel will update the *Intel® 64 and IA-32 Architecture Software Developer's Manual, Volume 3A: System Programming Guide* in the coming months. Until that time, an application note, TLBs, Paging-Structure Caches, and Their Invalidation (http://www.intel.com/products/processor/manuals/index.htm), is available which provides more information on the paging structure caches and TLB invalidation

In rare instances, improper TLB invalidation may result in unpredictable system behavior, such as system hangs or incorrect data. Developers of operating systems should take this documentation into account when designing TLB invalidation algorithms. For the processors affected, Intel has provided a recommended update to system and BIOS vendors to incorporate into their BIOS to resolve this issue.

§

Documentation Changes

There are no Documentation Changes in this Specification Update revision.

§